UNIVERSITY OF SOUTHAMPTON
SCHOOL OF ENGINEERING SCIENCES

MULTI-AGENT CONTROL FOR SPACE BASED INTERFEROMETRY

PRODUCED BY

N.K.LINCOLN

THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

AUGUST 2009

ABSTRACT

Agent systems have been accepted and used advantageously by computer scientists since their inception, but such systems have not been used so readily within the realms of control engineering or robotics. The work contained within this thesis investigates separated spacecraft interferometry in the context of a multi-agent system, under the influence of libration point orbital dynamics. The main focus is on the development of key agent skills, including state estimation, guidance, control and decision methods to attain the desired system output; within the consideration of decision methods, a comparison between centralized and distributed decisions is made. Whilst mainly focussing on the development of these skills, additional considerations pertinent to agent system development are also discussed.

A discrete time control method, integrating Kalman filtering with sliding mode control and using potential function guidance to achieve velocity tracking with six degrees of freedom, is developed for the purposes of controlling agent motion. Whilst developed for the purposes of spacecraft agent control, the presented methods are equally valid to any other vehicular agent system such as UAVs or AUVs if considering inter-agent regulation.

Centralized and distributed decision methods are developed to enable appropriate autonomous actions to be performed by the agent system. Primarily these actions include selective attainment and regulation of a non-natural orbits relative to a central agent to form an appropriate array configuration and instances of array reconfiguration to compensate for both failed agents and to maximize the mission duration.

DEDICATION

I am eternally grateful to my friends and family who provided endless encouragement and support throughout the undertaking of this work, and who continue to do so. The completion of this thesis is a tribute to you all. Thank you.

ACKNOWLEDGEMENTS

The author was first introduced to the subjects of satellite formation flying and autonomous control systems by Professor S.M. Veres at the University of Southampton. During a work placement at the European Space Research and Technology Center (ESTEC), under the guidance of Dr's C. Bramanti and D. Izzo, the author was involved with the theoretic application of distributed spacecraft systems, primarily that of interferometry, from which the topic of this thesis stems. Subsequent to the placement, the work initiated at ESTEC was continued under the supervision of Professor S.M. Veres, wherein aspects of guidance, control and decision making were investigated more thoroughly. The author would like to extend his thanks and gratitude to the aforementioned supervisors who have afforded countless amounts of assistance during the undertaking of this work, in particular to Professor S.M. Veres who provided invaluable assistance throughout the continuation and completion of this work. The author would also like to acknowledge the assistance afforded by Professor J. Fliege, from the University of Southampton, for an introduction to AMPL that was used to form (centralized) optimal solutions required within the presented work.

DECLARATION OF AUTHORSHIP

I, N.K.Lincoln, declare that the thesis entitled *Multi-Agent Control for Space Based Interferometry* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University;
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- Where I have consulted the published work of others, this is always clearly attributed;
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- Parts of this work have been published as:
  - Lincoln, N.K. and Veres, S.M. (2006). Components of a vision assisted constrained autonomous satellite formation flying control system. In *International Journal of Adaptive Control and Signal Processing*, $21(2-3)$,$237-264$.
  - Lincoln, N.K. and Veres, S.M. (2008). Six Degree of Freedom Variable Hierarchy Sliding Mode Control in Halo Orbits with Potential Function Guidance. In *Proceedings of the* 47*th IEEE Conference on Decision and Control*. Cancun, Mexico, Dec. $9-11$, 2008.

– Lincoln, N.K. and Veres, S.M. (2009). Discrete Time Sliding Mode Control for Satellite Clusters at Lagrange Points. In *Proceedings of the European Control Conference* 2009-*ECC'*09. Budapest, Hungary, Aug. 23 – 26, 2009.

signed: .....

date: .....

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AAP	Agent Array Point
ACL	Agent Communication Language
AI	Artificial Intelligence
AIL	Agent Infrastructure Layer
APL	Agent programming language
AUV	Autonomous Underwater Vehicle
AMPL	A Mathematical Programming Language
BDI	Beliefs, desires, intentions
CHARA	The Center for High Angular Resolution Astronomy
CI	Computational Intelligence
CPDA	Centralized planning with distributed action
CRTBP	Circular Restricted Three Body Problem
DAI	Distributed artificial intelligence
DOF	Degrees Of Freedom
DPA	Distributed planning and action
DSN	Deep Space Network
DTSMC	Discrete Time Sliding Mode Control
EKF	Extended Kalman Filter

ERTBP Elliptic Restricted Three Body Problem

ESA European Space Agency

ESTEC European Space Research and Technology Center

FIPA Foundation of Intelligent Physical Agents

HST Hubble Space Telescope

IATA International Air Traffic Association

ICE International Cometary Explorer

IMAS Intelligent multi-agent system

ISEE-3 International Sun/Earth Explorer 3

KF Kalman Filter

LMS Lefferts/Markley/Shuster

MATLAB Matrix Laboratory

MAS Multi-agent system

MEMS Micro Electro-Mechanical System

MTF Modulation transfer function

MIP Mixed Integer Programming

NASA National Aeronautics and Space Administration

NLP Natural language programming

NLS Nonlinear Least Squares

OASIS Optimal Aircraft Sequencing using Intelligent Scheduling

PRS Procedural reasoning system

PSSI Physically Separated spacecraft interferometry

RA Remote agent

RFF remaining fuel fraction

SAS Single-agent system

SFF Spacecraft formation flying

SMC Sliding Mode Control

SNR Signal to noise ratio

SO3 Special Orthogonal Group In 3 Dimensions

STM State Transition Matrix

TPF Terrestrial Planet finder

UAV Unmanned Air Vehicle

VLTI The Very Large Telescope Interferometer

VSC Variable Structure Control

NOMENCLATURE

α	Azimuth angle
β	Elevation angle
$\boldsymbol{\eta}$	Gaussian white noise
$\boldsymbol{\phi}$	State transition matrix
$\boldsymbol{\Upsilon}_i$	Potential function guidance vector, relating to i
ΔV	Change in velocity
F_h	Inertial Heliocentric Frame
F_l	Lead Agent Frame
F_{nb}	Agent Body Frame
F_w	Agent World Frame
$\hat{\mathbf{s}}_{ij}$	Relative position vector estimate for agent i to agent j
λ	Wavelength
Λ_m	The set of optimal AAPs relating to m array points
$\lfloor \boldsymbol{\kappa}^{\times} \rfloor$	Antisymmetric cross product matrix, operating on vector $\boldsymbol{\kappa}$
ω	Angular velocity
τ	Torque vector
$\mathbf{b}_{gyro}$	Gyroscopic bias vector
CM	Cost matrix

$\mathbf{C}$	Direction cosine matrix
$\mathbf{d}$	Interagent distance
$\mathbf{f}$	Force vector
$\mathbf{H}$	Measurement matrix
$\mathbf{I}_{(\mathbf{m},\mathbf{n})}$	Identity matrix of size $m \times n$
$\mathbf{J}$	Agent inertial matrix
$\mathbf{K}$	Kalman filter gain matrix
$\mathbf{P}$	Kalman filter state covariance matrix
$\mathbf{q}$	Quaternion vector
$\mathbf{Q}_d$	Noise covariance matrix
$\mathbf{q}_e$	Quaternion error representation used within EKF derivation
$\mathbf{u}$	Composite force and torque vector
$\mathbf{u}^*$	Ideal composite force and torque vector
$\mathbf{v}$	Interagent velocity
$\mathbf{X}$	State vector for the CRTBP
$\mathbf{x}$	Agent state vector
$\otimes$	Quaternion multiplication operand
ρ	System mass ratio for the CRTBP
$\Sigma_i(k)$	Compound discrete time sliding surface of agent i at time instant k
$\sigma_i(k)$	Discrete time sliding surface of agent i at time instant k
$\sigma_i(t)$	Continuous time sliding surface of agent i at time instant k
υ	Fill factor of a spare aperture array
ε	Positive definite scalar constant used within power rate reaching law

ς	Scalar constant used within power rate reaching law
$\widehat{x}$	Estimated value of variable x
$\widehat{x_k^+}$	A priori estimated value of variable x for time instant k
$\widehat{x_k^-}$	A posteriori estimated value of variable x for time instant k
$\widetilde{x}$	Measured value of variable x (subject to measurement error)
a_p	Scalar weighting for potential function guidance vector
A_x	X amplitude of halo orbit
A_z	Z amplitude of halo orbit
B	Interferometer baseline distance
b_p	Scalar weighting for potential function guidance vector
c_p	Scalar weighting for potential function guidance vector
D	Filled telescope optic diameter
G	Universal gravitational constant
h	Discrete time sampling period
L_x	Libration point x within the CRTBP
m	Agent mass
M_x	Mass of primary body x within the CRTBP
N	Number of agents
R_{filled}	Resolution of a filled aperture
r_i	Variable weighting parameter for potential function development, relating to agent i
R_{sparse}	Resolution of a sparsely filled aperture
t	Time

U	Pseudo-potential function for the CRTBP
X	X direction for the CRTBP
x_i^d	Desired value of variable x, relating to agent i
x^e	Error relating to variable x
Y	Y direction for the CRTBP
Z	Z direction for the CRTBP

1. INTRODUCTION

Within the computing domain, the paradigm of multi-agent systems has seen a recent explosion in interest as a means of solving massively parallel problems, in a computationally efficient manner. This paradigm of multi-agent systems has crossed over into the fields of robotics and logistics, as a means of accomplishing more effective and robust solutions to highly complex problems.

This thesis aims to investigate the application of multi-agent system principles to a space system. Of interest are the potential benefits availed to a system through distributed mechanisms synonymous with multi-agent approaches and how these compare to more traditional, centralized, approaches. The chosen system is that of physically separated spacecraft interferometry. An interferometer is naturally expressed as a distributed system, the components to which must interact in a strictly controlled and precise manner in order for meaningful results to be extracted. In addition to these requirements, a space based mission faces additional issues such as the inherent need for autonomy and robust operation. The design of space based interferometers is certainly not a new concept, with solutions being presented in the form of tethered and free-flying elements; however aspects of controlling a naturally distributed system rely heavily on traditional control methods and it is these which are sought to be replaced.

1.1 Structure

The scope of this thesis is quite large and consequently the thesis is organized into four parts:

Part I contains a review of background material of relevant to this thesis, including literature on multi-agent systems, interferometry and separated spacecraft control. Part I concludes with an overview of the problem to be investigated and envisaged operational situations that must be dealt with.

Part II encompasses mathematical material relating to the restricted three body problem, motion dynamics and guidance methodologies. It does not contain any novel material, but serves as a foundation for material presented within Part III: the restricted

three body problem environment is the dynamic regime to be used within simulation; motion dynamics and guidance methodologies, as well as aspects of the restricted three body problem material, are used within the development of all agent skills.

Part III constitutes the main contributions of the thesis, primarily developing the agent skills of state estimation, control action and decision methods. Part III draws upon information presented within Parts I and II, and is concluded with an analysis of the developed agent skills and a discourse of considerations relating to multi-agent systems design.

Part IV contains appendices to the thesis and primarily serves to contain material deemed too distracting for inclusion within the main thesis matter.

1.2 Publications During Work Undertaken

During the undertaking of the work presented within this thesis, key components have been presented at international conferences and within internationally accepted journals.

Journal publications:

- Lincoln, N.K. and Veres, S.M. (2006). Components of a vision assisted constrained autonomous satellite formation flying control system. In *International Journal of Adaptive Control and Signal Processing*, $21(2-3)$,$237-264$.

Refereed conference publications:

- Lincoln, N.K. and Veres, S.M. (2009). Discrete Time Sliding Mode Control for Satellite Clusters at Lagrange Points. In *Proceedings of the European Control Conference* 2009-*ECC'*09. Budapest, Hungary, Aug. $23-26$, 2009.

- Lincoln, N.K. and Veres, S.M. (2008). Six Degree of Freedom Variable Hierarchy Sliding Mode Control in Halo Orbits with Potential Function Guidance. In *Proceedings of the* 47*th IEEE Conference on Decision and Control*. Cancun, Mexico, Dec. $9-11$, 2008.

Additionally, work was completed resulting in contributions to the following conference publications:

- Veres, S.M. and Lincoln N.K. (2008). Sliding Mode Control for Agents and Humans-the use of sEnglish for publications. In *TAROS* 2008*, Towards Autonomous Robotic Systems*. Edinburgh, Scotland, 3 – 4 Sept. 2008.

- Veres, S.M., Lincoln, N.K. and Gabriel, S.B. (2007). Testbed for Satellite Formation Flying Control System Verification. In *AIAA InfoTech in Aerospace* Ronhert Park, CA, USA.

- Veres, S.M. and Lincoln, N.K. (2006). Vision assisted satellite formation control. In, *CDC 2006: 45th IEEE Conference on Decision and Control*, San Diego, USA, 13 – 15 Dec. 2006, 5712 – 5717.

- Veres, S.M., Lincoln, N.K. and Gabriel, S.B. (2006). Facility for satellite formation flying control system verification. In, *9th International Workshop on Simulation for European Space Programmes - SESP 2006*, Noordwijk, The Netherlands, 6 – 8 Nov. 2006.

1.3 Contributions of the Thesis

This thesis has contributions relating to state determination, control methodology and decision making.

State Determination The state determination method, based upon the use of a vision system to develop a local coordinate system within which the multi-agent system operates, is an extension of work within the field. Multi-agent system members cooperate to permit maximum likelihood estimates of position information and this is integrated into existing state determination methods.

Control The control method, that of discrete time sliding mode control, is a natural extension of existing work within the field. Discrete time sliding mode control is an established control technique, and although here the application of a discrete time exponential reaching rate law is advocated, a key aspect is the fusion of the control regime with Kalman filtering methods.

Decision Making A major contribution of this thesis relates to both central and distributed decision mechanisms that drive a space based interferometric system. Within the literature review it was observed that neither contingencies in the event of interferometer element failure, nor methods for prolonging an interferometry mission using non-natural orbits at a libration point orbit, have been considered. Both of

these points are addressed within this thesis, using both centralized and distributed methods. Additionally, both centralized and distributed methods are applied to the developed scenario to enable a comparison of the resultant system behavior due to each presented method.

Part I

BACKGROUND LITERATURE

2. MULTI-AGENT SYSTEMS

This Chapter will provide an introduction to multi-agent systems, their definitions (both formal and linguistic), design methodologies and current multi-agent system applications. It draws from literature not only existing within the realms of computer science, where the concept originated, but that within robotics and includes a Section introducing swarm intelligence, a trait exhibited by biological systems. The Chapter culminates with a brief review of current research into space based multi-agent systems and conceptual space missions within which multi-agent systems may be applied.

2.1 Agent Systems

A computational agent is a process which executes in order to perform a particular task and achieve a desired response; the agent is beyond that of a functional process in that it exhibits some degree of autonomy within operation. A single agent system (SAS) is a system controlled through a single process: this single process must consider all system entities, including input and output relationships. If operating within a large and complex environment, the resultant SAS will be equally complex in design; it may be beneficial to use multiple simpler agents to perform the same task.

A multi-agent system (MAS) is an artificial or biological system that is composed of multiple interacting agents: here the task of a single agent system is broken down across multiple agents such that each component agent is simpler in design. The intended action of a multi-agent system is to achieve a desired global objective, brought about from the cooperative action of individual agents: each only possessing partial system information and none having overruling ability within the agent community. Conceptually this can be viewed as global system output being the result of team effort rather than monolithic action. The aim of multi-agent systems research is to formulate methods of developing such complex systems of agents that are capable of achieving the desired global system behaviour.

The implementation of a multi-agent system is somewhat domain dependent. The suitability of multi-agent system implementation is addressed within [1, 2] wherein ap-

plication scenarios are suggested:

Dynamic or Complex Environments In such environments, it may only be possible to arrive at a solution through the use of a multi-agent system, since flexible autonomous action is required and these are properties synonymous with MAS. Agents allow for the handling of complexity by their natural modularity and abstract nature and so offer the benefit of programming simplification. This benefit is highlighted within [3], where systems such as those observed in process control are nominated as target applications, since they are inherently difficult to design and implement.

Natural MAS Environments For environments that are naturally modelled as a competitive or cooperative society, the application of a MAS is most natural. Wooldridge refers to this application area as *agents as a natural metaphor* [1].

Distributed Environments Environments that exhibit a high degree of distribution in data, control or expertise may not permit a centralized solution, yet such systems may be adequately modeled as a multi-agent system.

Legacy Environments Obsolete, yet functionally required, software can be reliably interacted through an 'agent layer', allowing for continual usage without the need for rewriting.

The MAS research field is a comparatively young area, having originated as a subfield of computer science during the late eighties; consequently the majority of literature has a strong bias towards this discipline. It was not until much later, in the late 1990*s*, that the research field took off into multiple directions, where the agent systems concept was expanded into fields such as robotics and complex biological systems.

The first MAS were software based, without physical processes of the environment. Later these software agents were deployed in some form of physical sensor and actuator hardware. This hardware may interact with another hardware system which itself may or may not have an agent software system deployed. With the advancement of robotic systems, teams of robots are being considered as a MAS, independent of the existence of agent based software running within the robot systems. Such a formalism retains the notion of a MAS being a system of multiple interacting agents: here the robots themselves represent the agents. However this current formalism would entail that the terminology of 'agent system' can be applied equally to an agent based software environment or a physical agent system, provided that the system being considered satisfies certain requirements, as prescribed below.

Subsequent usage of the term 'agent' will not directly refer to a software or hardware agent. However, an 'agent system' will be understood strictly within the software

domain, within which all decisions and computation is carried out. An 'agent system' can be attributed with an 'agent body', which is a physical entity and exists within the 'environment'. It is through this 'agent body' that the agent system interacts with the 'environment' within which other 'agent bodies' may exist.

2.2 Linguistic Definition of a MAS

The early definition of a software agent, provided by Wooldridge and Jennings in [2], has been widely adopted by the computer science research community for generalizing a software agent as *an entity which is situated in some environment, and that is capable of autonomous action in this environment in order to meet its objectives*. This is only one of many definitions of an agent system: [4, 5, 6, 7, 8, 9] provide additional descriptions, primarily generated from examples specific authors applied within their research. Whilst the description within [2] is highly generalized, the notion of an agent is captured entirely, and from this high level definition various subsets of agent systems can be identified. The most evident of these subsets of agents is that already introduced: a MAS. Within [1, 3] a distinction is made between MAS and *intelligent* multi-agent system (IMAS): a MAS is considered an IMAS when the agents composing the system are themselves intelligent. Within this development, *intelligent* was used to describe the ability to produce an appropriate response based upon current percepts, which implies rationality rather than cognitive intelligence. The proposed distinction between a MAS and IMAS, as suggested within [1, 3], is rarely used in practice since such a division would negate the existence of intelligence as an emergent trait of certain agent behaviors and their resultant interaction. Whilst reference to an IMAS is rare, the concept of a specifying use of rational or behavioral agents has become favored by the community as a means of agent system specification [8, 9, 10].

Whilst numerous specialized agent types have been defined within literature, the subsequent discussion will only be concerned with agents within the context of multi-agent systems and not the broad spectrum of individual agents which exist; for such details the interested reader is referred to the current literature [1, 8, 9, 10].

For classification as a MAS, the agents comprising the system may have the following characteristics:

Situated An agent must be situated in some environment. This is a very light constraint: the environment can be purely software based or may include physical processes other than those of digital hardware. It is however the nature of this environment that is of interest to an agent system, in particular a MAS designer, and this must be formalized in some way.

Autonomous Each agent must be capable of at least some partial autonomous action: the ability to decide for themselves what action is required in order to satisfy their design objective(s) within the environment they exist. Some authors are reluctant to use the term 'autonomous', preferring 'rational' to describe agents that take goal oriented action based upon percepts and knowledge of the inhabited environment [6]. Regardless of the exact terminology used, in a computational sense, this is what distinguishes agents from objects.

Perception An agent must be capable of sensing it's environment, since this is the key to autonomous action. To echo the distributed nature of MAS, within [1] it is stated that each agent operates based upon local knowledge; no agent has a global view. This can equally be considered as a single agent not being able to make practical use of the global knowledge, even though this knowledge may be available. Arguments can easily be made that a MAS could work more efficiently if each agent possessed global system knowledge and this is evident within hierarchical multi-agent systems which are used to coordinate agent activities [11].

Social Each agent is capable of interacting with other agents. The term 'interaction' here relates not only to a message passing ability, but other interactions such as coordination, negotiation, etc. This notion ties together the requirements of autonomous and decentralized operation: it is through this interaction process that the ability of an agent system to achieve a global operation based upon local knowledge is realized.

Active This concept comprises of two key activities: reactivity and proactivity. A strictly reactive agent has an idle state wherein it waits for a triggering event before commencing any form of action, much like an object. Conversely, a proactive agent does not have an idle state and continually performs some form of action. Strictly reactive or proactive agents both have their advantages: a reactive agent consumes less resources and does not require 'training' as implementation is by direct condition-action. However, in time critical tasks a proactive agent will have all pertinent information to hand and thus the ability to solve issues in a more time efficient manner. Moreover, a well configured *intelligent* proactive agent would be able to foresee and deal with problems before they develop to the stage of triggering a reactive agent.

There is an obvious benefit in agents which are heterogeneous in activity, being both reactive and proactive. When formulated correctly such a combination introduces a robust nature to the system. A heterogeneously active agent is proactive in the

persistence of attaining system goals in a slowly variable environment and reactive to sudden changes in the perceived environment within which the agent is situated.

There are two characteristics which are slightly more controversial in nature when applied to the realm of MAS: decentralized and intelligent. Whilst in [1] it is stated that no agent should have direct control over another agent, since this would then reduce the system to a monolithic architecture. There is a caveat that an agent is permitted to *influence* the decision of another agent, but it may not prescribe the decision: it is this caveat to which agent system design most closely adheres to, especially those for supervisory and hierarchical agent structures, as presented within [12]. Intelligence is a highly contended term when applied within the context of computation, immediately entering the fields of artificial intelligence (AI) and computational intelligence (CI). Remaining strictly within the agent systems domain, and certainly not entering a philosophical discussion on the ability of an electronic system to exhibit a biological trait, quantification of what constitutes an intelligent agent can be considered as the ability to learn and apply knowledge. Applied to a system this entails the ability to accommodate new problem solving skills which may be self generated, to perform self analysis and modify both the skills and future actions based upon this analysis.

As is already evident, the multi-agent system world is abundant with metaphorical expressions which give living analogs to the artificial agents within a system. It is perhaps prudent here to emphasize that agents are computational, information processing entities: metaphorical analogs are only intended to convey high level concepts.

The usage of concepts and human oriented metaphorical expressions for the desired action of a computational agent is itself perhaps an indication in the paradigm shift of current programming methodology: that of increasing locality and encapsulation. Considering the scenario of software programming, the original unit of software was a complete program which later evolved into 'structured programming' where the software was formed by smaller packages of code; each subroutine being self contained and invoked by being called externally. Structured programming gave way to object-oriented programming, conceptually similar to the structured methods but each object[1] can interact in a cooperative manner as opposed to computing a structured set of tasks: objects are capable of receiving, processing and passing data to other objects. Object-oriented programming focuses on data rather than processes: an object-oriented program consists of multiple objects, each containing all the information needed to manipulate its own data structure. An agent oriented program takes the object oriented methodology a stage further in that the decision to implement a particular piece of code is made by the agent itself; interaction is by choice. In addition, each agent (classifiable as a special class of object) has its

[1] In its simplest form, an object can be defined as an allocated region of storage.

own 'goals' and resultantly not only is the invocation localized, but the code and data is too. This progression of programming structure has been shadowed by the methods of human-computer interaction: initial (primitive) interaction was through the use of physical switches, which gave way to command line interfaces and the rise of graphical user interfaces, to which object oriented programming is so well suited.

Emergent programming techniques include agent programming languages (APL) and natural language programming (NLP). These techniques directly encode software with the ability to utilize the metaphorical expressions widely used within the agent community.

APL is similar to object-oriented programming, however with APL the objects (now termed agents) decide when to run; there is an autonomous aspect to their invocation. Wooldridge surmises the difference between agents and objects as *'Objects do it for free; agents do it because they want to'* [1]. Within APL, the agents are directly programmed in terms of mental notions, developed by agent theorists to represent properties of an agent such as their beliefs, desires and intentions, resulting in a BDI model[2]. Such an abstraction combines the ability to compute with human notions which may be beneficial to express the operation of a complex system.

NLP is a software writing methodology using a natural language which is machine readable, yet equally readable by a human. NLP should not be confused with computational linguistics, which is mainly aimed at understanding a language to enable rapid translation of scientific journals, but as a method of simultaneously forming a computational and human language in the context of the problem at hand.

2.3 Formal Definition of a MAS

Although the literature regarding agent systems is abound with metaphorical expressions, as would be expected from a topic emergent from the realms of computer science, a formal high level definition of an agent system has been proposed within[13]. Whilst applicable to some software *or* physical agents, the formal definition presented within [13] is not universally applicable to *all* software and physical agents: for this thesis the following universal semi-formal definition is presented:

MAS We can describe a multi-agent system as a three-tuple, consisting of the set of agents, an environment in which the agents exist and a coupling between the agent and its environment.

$$MAS = < Agents, Environment, Coupling >$$

[2] BDI models are expanded upon in subsequent Sections.

Agent We can describe the i^{th} agent from a set of n agents as a five-tuple, consisting of the agent state, input, output, process and an agent body.

$$Agent_i = < State_i, Input_i, Output_i, Process_i, Body_i >$$

within this context, the state is a set of values which completely define the agent; the agent state itself is changed by the autonomous process. The input and output are themselves subsets of the state which are coupled to the environment through the agent body: hence, an agent is linked directly to the environment and indirectly to other agents.

Environment Formally we can describe the environment within which an agent exists as a three-tuple consisting of the environment's state, the processes occurring within it and all agent bodies.

$$Environment_e = < State_e, Process_e, Body_e >$$

within this context it is implied that the environment is dynamic, possessing its own processes which change the environment state, independent of agent body influences. Unlike the definition of an agent, there is no input/output, which implies that the environment is unbounded.

The semi-formal definition given above, modified from [13] and applicable to both software and physical agents, does not in any way conflict with the linguistic description of an agent suggested by [1, 2, 4]. The linguistic descriptions offer a conceptual method of explaining the notion of an agent system, whilst the semi-formal method gives an explicit description of the agent system and its interaction processes. These definition methods should be considered in conjunction, rather than individually, as independently they offer only a partial insight.

2.4 Agent System Design

MAS design inherently gives rise to two distinct, yet dependent, problems: those of agent design and agent society design, sometimes referred to as a micro/macro distinction. The challenge within MAS research is not only in breaking down the description of an agent system into what individual agents should do, but the implementation of a suitable architecture to achieve this. Solutions are generally sought using the well proven tools from game theory, economics and biology, which are themselves supplemented with concepts

from artificial intelligence research such as planning, reasoning methods, search methods and machine learning. There are three agent models which have gained a significant attention within the field

Agent Utility Model The utility model of an agent is perhaps the most popular theoretical model, possibly due to to the associated flexibility and roots in game theory and artificial intelligence. Within this model format, an agent is constructed as a utility maximizer inhabiting a Markov decision process[3] [1, 5]. The utility function[4] is used to capture an agent's preferences by providing a map from the states of the environment, or outcome of a game, to a real number that indicates how good a particular outcome is for an agent. Such an implementation perfectly expresses a self interested agent and naturally leads to a preference ordering over outcomes: the greater the utility, the more preferred the outcome. The mapping process occurring within a utility function can be viewed as an agent policy; this policy may change throughout time as a result of agent (cf machine) learning processes. Such agents are inductive in nature, since they extrapolate conclusions from presented evidence.

Logical Agents The modeling of agents as logical inference machines is favoured by scientists working on semantic or logical applications of agents. Logical agents are inherently deductive, since they deduce facts based upon rules of logic. [1] presents an overview on logics for multi-agent systems, largely focussed towards epistemic logics[5] and possible worlds semantics. Temporal logic[6] is perhaps the most suited form of logic system to permit agents to act in a purely deductive manner and has been prototyped within Concurrent MetateM [14, 15, 16]. In addition to allowing for agent programming, computational logic provides a well grounded and rigorous framework to study the syntax, semantics and procedures of agent tasks. Agent interaction can also be addressed using the same tools, thus linking the specification and verification of properties related to individual agents and agent communities.

BDI Agents The Belief-Desire-Intention (BDI) agency model is perhaps the most widely used model for practically reasoning agents. A BDI model implements the principal aspects of Bratman's theory of human practical reasoning [17] as a method of

[3] A Markov decision process is one in which the outcome of a decision is partly random and partly under control of the decision maker.

[4] In game theory this is referred to as the Von Neumann-Morgenstern utility function.

[5] Epistemic logic is a subfield of modal logic and concerns reasoning about knowledge

[6] Temporal logic is used to describe any system of rules and symbolism for representing and reasoning about propositions qualified in terms of time

explaining future directed intention. Beliefs[7] represent the agent's knowledge of the world: what it believes about the state of the environment and itself. Desires encapsulate the objectives of the agent, or equally the situations which the agent would like to bring about. Some authors equally refer to desires and goals, however usage of goals requires consistency whereas desires may be disparate. Intentions represent the course of action that an agent has chosen to take and can be seen as a desire which the agent has selected to fulfil. Although not explicitly mentioned in the model name, plans are equally important to complete the agent model: a plan is a sequence of actions which can achieve the current intention and may itself be to plan a course of action. Numerous development languages for BDI models already exist, examples being AgentSpeak, 3APL, JADE, JASON and JACK.

The Procedural Reasoning System (PRS) is a situated real time reasoning system which uses a BDI agent architecture model. Within this model exists a knowledge library containing a database of procedures that will accomplish particular tasks. This knowledge library is interfaced by an interpreter, which is also linked to the agent's BDI architecture. The interpreter (or reasoning system) is responsible for keeping track of the agent's BDI state whilst selecting the appropriate knowledge area to apply to the current task. A PRS system does not generate a complete plan of action and replan upon unexpected events; planning and action are concurrent activities.

None of these models expressly indicate the method or extent of communication between agents or the application variance between hetero or homo-dynamic systems. Although it is evident that with increased communication, more complex behaviors within robotic multi-agent systems are possible [18]. Within software multi-agent systems (where the size of an agent is arbitrary) complex behaviors with minimal communication can be achieved through further division of the agent.

2.5 Swarm Intelligence

Swarm intelligence is a concept directly linked with the notion of a multi-agent system. It can be defined as a form of emergent artificial intelligence based upon the collective behavior of decentralized and self organized systems: multi-agent systems. In a biological context a swarm is used to describe a large group of similar sized animals which are in aggregate motion. The number of entities required for a group to be classified as a swarm

[7] Here the notion of *belief* allows separation from knowledge and thus permits an agent belief to both be incorrect and change in time.

is not specified, though it is indirectly implied that the transition is made when the total number of biological entities is 'large'. Due to this ambiguity in terminology, a multi-agent system will not be referred to as a swarm, though a multi-agent system is assumed to be capable of exhibiting swarm intelligence.

Perhaps the most visual emulation of swarm intelligence exhibited by a multi-agent system is presented within the context of swarm robotics[8]. This scenario involves a large number of robotic entities, wherein each robot is considered an agent body, and these bodies may or may not be either homogenous or intelligent. These robots exist in some dynamic environment with which they interact via their outputs and which are themselves dependent upon the robot percepts, state process and body type[9]. The physical dimensions of the robot agents are not important[10]; it is the autonomous operation of the agents within the robot swarm and the local interactions made to effect a desired global outcome that are pertinent. The main considerations of such decentralized autonomous robotic systems are the same for general multi-agent systems: that of individual agent planning or decision making, consequent agent group dynamics and the evolution of agent behavior, which in actuality is the study of swarm stability.

2.6 Inspiration from Nature

Biological systems perfectly capture the essence of a multi-agent system: successful biological systems are composed of self organized basic entities and are capable of dynamic adaption to changing circumstances without any form of top-down control. As a consequence, such systems are highly studied by biologists and computer scientists alike. At its most ambitious, a multi-agent system aims to replicate this emergent cooperative phenomena that is typified by biological systems.

The evaluation of certain aspects of any biological system is very subjective and in part depends on the scope of the system which is to be considered. It is common to break a system into elements that are of interest and evaluate these, rather than the system as a whole.

[8] Swarm robotics could more appropriately be named 'multi-agent robotics', though the current literature prefers the prior terminology.

[9] This description precisely fits the formal definitions of a MAS, agent and environment provided within Section 2.3.

[10] Within [19], the notion of swarm robotics to form a Turing machine is presented, as is the use of larger swarm robots for factory purposes.

2.6.1 Ants as Agents

Possibly the most studied and theorized of biological multi-agent systems are those of ant colonies. An ant fits our description of an intelligent agent: it is situated, autonomous, active, social and acts entirely based upon local perception. In addition, an individual ant is not particularly capable: left to its own devices an ant would be able to achieve very little and most likely perish; yet a group of such ants can thrive, even in a hostile desert environment. The primary elements of focus to scientists studying ant systems in the mid nineties included efficient route formation and brood sorting; both of these are emergent traits evolving globally from very simple individual behavior.

Ants are capable of forming and subsequently refining routes to food sources by subjectively depositing time-decaying pheromones [1, 13, 20, 21]. This is a 'recruitment' process resulting in what would be described mathematically as minimum spanning trees; yet it is obvious that a simple ant is unable to apply conventional (human) graph theoretic computation algorithms.

The common model used for the ant is that the ant wanders randomly, executing Brownian motion. If a pheromone is sensed then this Brownian motion is influenced, such that the stronger the pheromone scent, the stronger the influence to follow in the direction of the scent. The pheromones act as a form of recruitment, encouraging more agents to follow in their direction: hence the terminology applied earlier. Should the ant come across some food source, the ant will pick up the food[11] and then attempt to find the nest. Upon acquisition of a food source, the ant will drop a time-decaying pheromone at a constant rate as it moves. There are now two possible eventualities, which are equally valid: either the ant follows in the direction of a separate pheromone belonging to the nest, or continues the random walk influenced by the same pheromones sensed previously (if any exist). Regardless of the method implemented to search for the nest, at some point in time the ant will encounter the nest, deposit the food and recommence the search for food. This simple model results in two observations. Firstly, there is the implicit suggestion that at some stage the ant will encounter both a food source and the nest when completing a random walk[12]. Secondly, there is an emergent behavior where the routes to a food source are refined, remain robust to disturbance and disappear when a food source is depleted. This occurs since only food carrying ants will drop pheromones and once a food carrying ant reaches the nest there will be a trail linking the food source with the nest. Initially, the trail will be far from nominal due to the random walking processes, but the random walking process is also what causes the pheromone trail to evolve into an efficient direct

[11] It is implied that the ant is capable of picking up a unit of food without assistance.

[12] As mentioned in [13], this assumption is perfectly valid if the nest-food distance is small in comparison to the range of the ant.

route: short cuts appear in the initial trail, which as noted within [20] will result in a faster build up of pheromones on the shorter route and thus enforcing its preference. In contrast the pheromones originally laid on the longer trail will fade and the path will cease to exist. The eventual result is a direct line from the nest to a current food source. Of conceptual interest is the time decay process involved with the pheromone deposits: considered in a physical context, once a food source is depleted the path becomes less traveled and so is eventually discarded entirely; applied to an AI context, we are reminded that the ability to forget is as important as that of remembering. These globally emergent traits, resultant from individual actions, have led to the development of so called 'Ant Colony Optimization' routines [20, 22, 23, 24] and in practical applications have also been used effectively within robo-soccer algorithms [25].

2.6.2 Other Biological Agents

Stigmergy[13] is evident within numerous biological systems and it is certainly not limited to the emergent behaviors observed within ant colonies. Indeed, the foundations of such biological studies were focussed on the cooperative action of termites observed building a nest, where it was observed that the nest structure forms initially as stacks, coalescing into a sequence of arches. Similar to the ant route formation model, termite nest construction can also be reasoned about by use of the pheromone concept. In this model the building blocks created by a termite are seeded with a time-decaying pheromone and randomly deposited in the area of a suitable nesting site[14]. As with ant motion, placement of a building block is influenced by pheromone presence, with the termite preferring to place a building block in proximity to a high concentration of pheromones. Although initially being placed randomly, at some stage the proximity of several building blocks will cause a preferential location building block placement: pheromone levels in less frequented areas will drop and so building blocks will only be deposited in specific locations resulting in the formation of towers. Due to the time decaying nature of the pheromones, there will be a pheromone gradient through the height of the tower with the greatest concentration being at the top. Neighboring towers will also be sensed by a termite and so the tops of the towers will grow towards each other, forming the characteristic arch shape. As with the ant example, no single termite decides how the structure is to be built, it simply evolves.

Moving away from attributes exhibited only within the insect world, emergence can be

[13] Stigmergy was originally defined by French biologist Pierre-Paul Grassé in 1959 as 'Stimulation of workers by the performance they have achieved'. Stigmergy is more clearly defined as a form of communication by modifying the environment and is now used to express the mechanism of indirect agent coordination where the trace of an action in the environment stimulates subsequent actions.

[14] The process for selection of a suitable nesting site is not considered here.

observed in the flocking behavior of birds, fish and mammals. Within flocking behavior, movements are coordinated in conjunction with the consideration of collision avoidance between other agents and obstacles in addition to the avoidance of predators. This occurs without any form of sophisticated communication or centralized coordination; global coordination is resultant from individual actions propagating through the system.

2.7 Implemented MAS

Multi-agent systems have already found their way into applications as wide ranging as air traffic control, industrial processes and spacecraft systems [1, 3, 26]. These applications will now be presented.

2.7.1 Traffic Management

The flow of traffic is inherently a multi-agent system, being both geographically and functionally distributed; it is equivalent to that of animal migration and flocking. In a traffic flow system, each human piloted vehicle can be considered an agent, with its own beliefs, desires and intentions.

Optimization of traffic flow is an important consideration since it has economic, environmental and social impacts. Within [27], a reference is made to a study on European air traffic congestion, prepared by SRI International for the International Air Transport Association (IATA) in April 1990, forecasting that in the year 2000 air traffic congestion alone will cost Europe US$10 billion annually in national losses due to constrained growth. Although such statistics are now significantly out of date, they do serve appropriately to highlight the economic impacts of non-optimal traffic flow and the magnitude of the problem. Obvious economic parallels can be drawn with both sea and land transportation congestion. Further economic and environmental issues arise in considering the requirement to minimize fuel usage in all forms of transport, a subject which is also becoming a political consideration, due to the growing popularity of environmentalist ideals.

In principle there are two methods of mitigating traffic congestion: increased infrastructure or increased flow efficiency. The former method entails the construction of more roads or rail networks, runways or ports; all of these options are ultimately constrained by physical space availability and a considerable capital investment. As a consequence, methods to increase the efficiency of traffic flow have received much interest.

Traffic management has traditionally been viewed as an optimization process and, at its most complex, involves intermodal transportation connectivity. Within [28], the implementation of agent based systems is examined for the problem of transportation manage-

ment and also details traffic research efforts made by Daimler-Benz. A key observation is that current logistical operations do not involve active planning, generally relying upon more traditional reactive methods.

Air traffic control in particular has received a large amount of attention, with agent based research dating as far back as 1986 [29]. Numerous theoretical papers have been published on the subject, with a concrete formulation being presented within [30]. Within this paper, artificial intelligence (AI) and distributed artificial intelligence (DAI) techniques are applied to air traffic flow management with the aim to maximize the utility of air traffic resources; such an implementation relates back to the notion of agents as utility maximizers. A practical implementation of an agent based system for air traffic control was successfully field tested at Sydney airport in Australia during 1995 [27]. The system, designed and implemented by the Australian Artificial Intelligence Institute, received the acronym of OASIS: Optimal Aircraft Sequencing using Intelligent Scheduling. The system was designed to give assistance to air traffic controllers through the provision of tactical air traffic management. Implemented using a PRS architecture, the system is heterogenous, being both proactive and reactive. The prototype OASIS system was intended to be succeeded by a commercial version named HORIZON, with an initial release date set for 1997. The roll-out of these systems appear to have come across some form of resistance since no recent publications regarding either system exist, aside from some unreliable internet sources indicating that public safety concerns, involving a reluctance to accept more recent and unproven technologies, have derailed the program.

Systems such as OASIS are not only of benefit to the air traffic sector, but would be of huge benefit to busy international shipping ports. These systems, although restricted to a 2D environment, are exceptionally complex not least due to the severely limited manoeuvrability of current super-tankers and equally of high risk due to the potential for ecological disaster, as evident from the MT Hebei Spirit oil spill in 2007 which resulted in the spill of $10,800$ tonnes of crude oil into the Yellow Sea [31].

2.7.2 Industrial Applications

Agent systems used within traffic control represents only one of many domains for which an agent approach is suitable[15]. The industrial sector, being intrinsically profit driven, has much to gain from the implementation of agent systems and consequently has applied them for both the design and operation of industrial systems. Application of agents systems within the industrial sector varies across all forms of distribution, from modeling to end-product: inclusive of emulation, prototype and pilot schemes.

[15] Here we refer back to the often quoted agent system applicability criterion: when a system is modular, decentralized, changeable and complex [1, 2].

Within [26], Parnuk details the numerous agent systems within industry and emphasizes that industrial systems are driven by the need to solve a practical problem, rather than curiosity in potential applications, as is exhibited in academic research. Perhaps the greatest insight within [26] is the need for standards and techniques within research fields that make applications of agent based techniques accessible to industrial users. It is not the purpose here to replicate the details of [26], but to convey to the reader the huge breadth of agent systems application within the industrial sector from new design methodology as shown by RAPPID[16], unique artificial intelligence $3D$ animation systems such as *Massive*,[17] to innumerable plant operation programs.

With the further development of robotic systems, swarm intelligence approaches have emerged within logistical applications. Distribution centers offer the ideal arena for implementing a multi-agent scheme: Kiva systems [32], have developed Kiva Mobile Fulfillment System (MFS) which is a multi-agent robotic system used for warehouse order fulfillment and other vertical market applications. Within the system, semi-autonomous robots are used for efficient order processing. The classification of semi-autonomous here arises because, although the robots are capable of autonomous processes such as recharging and route tracking, their tasks are assigned by a centralized computer. Nevertheless, the system is still classifiable as a multi-agent system. Logistically, for the MFS, inventory pods are stored in the center of a warehouse while operators stand at inventory stations around the perimeter. Upon receipt of an order, robotic drive units retrieve the appropriate inventory pods and bring them to the worker to collect the appropriate item. Completed orders are stored on separate pods, ready for transportation to the loading dock when a distribution vehicle arrives. Inventory pods do not have a fixed location, which is in stark contrast to the traditional warehouse organization architecture, where items are stored in precise locations on static shelving. The system allows for massively parallel processing of orders, a trait typical of a multi-agent system. As a result, the Kiva MFS has achieved breakthrough improvements over generally accepted material handling benchmarks, including order picking speed, order accuracy, volume scalability and flexibility in reconfiguring operations or responding to market changes[18][33].

[16] Responsible Agents for Product-Process Integrated Development has been piloted in the high-level design of a military vehicle at the U.S. Army's TACOM facility [26].

[17] Massive software has been used within numerous high profile films including 'The Lord of the Rings' trilogy, 'King Kong', '300', 'The Chronicles of Narnia: The Lion, The Witch and The Wardrobe', 'The Ant Bully' and 'I, Robot'. It is capable of rapidly creating thousands (or millions) of individual agents which can act individually; effectively creating and simulating a multi-agent environment.

[18] Staples Inc. and Zappos.com, a competitor to the more successful amazon.com, has chosen to implement the Kiva-MFS system having completed successful trials.

2.7.3 Spacecraft Control

The importance of introducing a virtual human presence in space is introduced within [34] and in particular highlights the need for increased autonomy to achieve such ambitions. There is an obvious risk in the usage of new technologies in space and, in an effort to mitigate this necessary risk, NASA launched a 'New Millennium Program' aimed at testing future technologies to aid development and increase the confidence of new systems. Deep Space 1, a spacecraft launched in 1998, was part of this program and among the 12 new technologies to be validated was a system known as Remote Agent (RA). RA is an autonomous agent based control system designed to "addresses the unique characteristics of the spacecraft domain that require highly reliable autonomous operations over long periods of time with tight deadlines, resource constraints, and concurrent activity among tightly coupled subsystems" [34]. RA is formed by the integration of three separate technologies: an on-board planner-scheduler named EUROPA, a robust multi-threaded executive, and a model-based fault diagnosis and recovery system called Livingstone. Instead of being commanded to execute a sequence of commands, the system design is such that the attainment of a list of goals is sought. From the specified goals, the RA forms a plan to accomplish the goals and then executes the plan, whilst maintaining feedback on the current plan execution and the current status of flight hardware. [35] details the RA design philosophy and also the scenarios intended during the testing phase of the system: power bus status switch failure, camera power stuck on, hardware device not communicating over bus to flight computer and thruster stuck closed. The Remote Agent ran the spacecraft computer for a duration of two days over 60,000,000 miles from Earth and although some anomalous behavior was detected, resulting in the termination of the experiment, the mission was considered a success [36].

The RA architecture failed to make full use of the agent system paradigm and was effectively a top layer installed ontop of the regular flight software; subsequent investigations into such control systems involve a less conservative, complete agent based system approach. [37] and [38] provide high level architectures for an agent based control architecture solution to satellite formation flying. The generalized architecture presented within [37] was destined to be adapted and tested on Techsat$-$21, a fleet of three low Earth orbiting spacecraft to be used for technology demonstration. Originally the Techsat$-$21 mission was planned to be launched in 2004, though the project was canceled in 2003 due to the technical difficulties involved [39, 40]. The agent architecture produced is nevertheless still valid and of interest: each satellite was to be programmed in a C++ implementation of what was termed ObjectAgent to result in a 'TeamAgent' system. The TeamAgent system was hierarchical, with one of the satellites acting as a cluster manager and performing all cluster-level computation relating to user specified configuration

commands. The architecture presented within [38] is programmed using Jack, whilst incorporating additional procedures to enable usage of MATLAB for numerical computation. The paper highlights performance requirements and suggests algorithms capable of achieving the desired agent based control, though alike [37], no specific control or communication routines are presented. Unlike [37] the destination hardware was not that of space: a ground based test bed had been build to test the agent system, successfully avoiding the problematic political and financial issues faced with space-based testing of hardware.

2.8 Space Based Multi-Agent Systems

The RA test, flown aboard the Deep Space One spacecraft, is an example of agent based software operating in a challenging environment where goals are to be achieved through the formation and execution of plans. In addition to agent based software techniques for individual spacecraft, space based missions can utilize a multi-agent systems approach to create distributed hardware systems[19]; essentially swarm robotics applied to space missions. Whilst the terminology of swarm robotics has just been applied to a distributed hardware space mission, the European Space Agency is developing an Earth Explorer mission named 'Swarm', consisting of three polar-orbiting satellites anticipated to launch in 2010 [41, 42]. To prevent any confusion between this mission and any conceptual space based multi-agent system, subsequent usage of 'swarm' will be omitted and 'multi-agent system' used for reference to any physical system composed of n elements, where $n > 1$.

Technological advances in micro-manufacturing processes are allowing for the design and fabrication of previously unimaginable satellite systems. With the emergence of functional nano-satellites[20] and current research being directed towards the possibilities of manufacturing pico-satellites[21], combined with the maturing field of satellite formation flying, the implementation of multi-agent systems in space is a real possibility. The implementation of such systems in the space environment would carry across all of the benefits associated with terrestrial applications:

- Current space hardware is susceptible to single point failures: satellite subsystems are not replaceable and should they fail then the mission is generally lost. Considering the application of multi-agent systems to this instance, this issue of survivability

[19] A distributed hardware system is also sometimes referred to as a virtual platform.

[20] Nano-satellites are classed as satellites not exceeding a wet mass of 10kg. Examples of nano-satellites which have already flown include SNAP-1, from SSTL [43], and Mini AERCam from NASA [44].

[21] Pico-satellites are classified as satellites with a wet mass not exceeding 1kg. A current example of a pico-satellite implementation is CubeSat [45].

is overcome in the same way that loss of a single ant does not endanger the colony: should any elements be lost, the the mission can continue with only some loss in performance. This is the well cited 'graceful degradation' concept.

- Multi-agent systems are inherently modular and this is echoed within the terrestrial swarm robotics field. Transfer of this modularity to a space system would permit adhoc replacement or upgrade of elements during the mission to enhance system performance or simply extend the mission life span. Additionally, by moving away from implementing a single monolithic entity, dimensional launch constraints are removed, presenting numerous advantages: current launch vehicles can be used without modification[22], the risk of launch can be mitigated by spreading the system across several launch vehicles and further mission savings can be made through possibilities of sharing a launch vehicle.

- MAS benefit economically since mass manufacturing processes allow for the cost efficient production of elements when considered on a price-per-piece basis. This economic aspect is directly transferable to the space sector, which traditionally implements multi-million $ monolithic platforms.

It is clear that the benefits presented cannot be ignored and certainly not underestimated. Once the formalities and technicalities of the required new technologies have been addressed, MAS will undoubtedly start to play a significant role in the future of space systems.

2.8.1 Conceptual Multi-Agent Space Missions

Whilst the potential of multi-agent spacecraft systems has primarily been championed through abstraction of terrestrial benefits, there are two core considerations relating directly to space based applications, which although briefly mentioned in previous Sections, are of direct relevance: scalability and autonomy. A perfectly scalable system is one whose performance increases linearly with the increase of system size. Perfectly scalable systems are perhaps the most suitable missions for the application of multi-agent systems, since increasing the community size will result in greater system performance. In areas requiring a high degree of autonomy, be it from lack of continuous communications resultant from LEO or mission critical communications time lag resultant from operating at extreme distances from Earth, the ability of autonomy as exhibited by RA is necessary.

[22] The launch of ENVISAT required the modification of an Arianne IV rocket shroud to accommodate the satellite.

With the benefits of applying multi-agent system concepts to the arena of space missions being evident and the technology required to achieve this being available, what is of interest is, perhaps, which missions are suited to such approaches.

Exploration The application of multi-agent systems to space exploration can yield huge benefits to exploratory missions in terms of scientific return. A single spacecraft is resource limited in exploring potential: the sensors can only be trained to a specific point in as much as the spacecraft can only be located at a single point at any one time. By use of multiple sensing devices, the rate of scientific return would be proportionately greater. This instigation of distributed measurement is a perfect application for the research being carried out within the e-CUBES project: MEMS[23] based 3D integrated micro/nano modules for easily adapted applications that have an inherent disposition towards distributed measurement systems [46].

A multi-agent systems approach for in-orbit planetary observation lends itself to two distinct operational possibilities, depending on the destination environment. In particularly harsh environments, such as Venus, where the lifetime of a probe is severely limited, continual launch of 'disposable probes' with over-lapping lifetimes would allow for a much longer duration mission. Alternatively, a large group of probes, homogenous or otherwise, could be launched simultaneously and descend through the planetary atmosphere making a series of concurrent measurements. This would enable a volumetric approach to atmospheric measurements, which could be of benefit to the scientific community. Moreover, instead of launching a series of probes, it is entirely possible that a fleet of planetary rovers (land or air based) could be dispersed, thus enabling the return of significant amounts of data over a prolonged period [47, 48].

A multi-agent mission would be of great benefit to asteroid or comet exploration: not only would the risks associated with flying in a dangerous environment be mitigated but, as with a swarm of planetary rovers, exploration would be more efficiently carried out, thus permitting faster scientific return. Very little is known about the asteroid belt located between the orbits of Mars and Jupiter: physical and geological data is severely restricted by the limited data obtainable from Earth-based observations, and only a few spacecraft have encountered the asteroid belt. To date, although over 45000 asteroids have been catalogued, millions are yet to even be discovered. Moreover, important physical parameters such as mass, bulk density, surface geology and composition can only be obtained by close proximity

[23] Micro Electro-Mechanical System is the integration of mechanical elements, sensors, actuators, and electronics on a common silicon substrate through micro-fabrication technology.

fly-by of a suitably equipped spacecraft. Asteroid belt exploration with a distributed platform has been proposed by both ESA and NASA within the APIES and ANTS programs respectively. APIES is a mission concept for using a fleet of spacecraft to study asteroids in the main belt between the orbits of Mars and Jupiter [49]. It is intended to release 19 Belt Explorer (BEE) satellites from a centralized mother-ship named the HIVE (Hub and Interplanetary VEhicle) to survey a minimum of 100 asteroids. When passing within a pre-defined proximity of an asteroid, the nearest BEE will move towards the asteroid and complete measurements of the asteroid mass, density and surface properties. All measured data will be relayed back to Earth via the HIVE.

Both exploration scenarios give us two insights to swarm robotics applied to deep space exploration:

- The operational principles of each MAS member can be best realized by use of agent programming techniques due to the nature of the environment within which they exist. This further supplements the abstraction of agent software and agent robotics and their interdependence.
- In all scenarios, the swarm members relay information back to a central mother craft which is used to transmit all data back to Earth. Although this does represent a single point failure within the system, enabling direct agent communication with Earth would severely constrain the design of roving agents.

Inspection Tasking Catastrophic incidents with the Space Shuttle have lead to a requirement of visual inspection of the shuttle exterior prior to de-orbit and landing to ensure that there are no potential problems which could endanger the lives of crew members. This process requires 18 man hours, with two crew members participating in Extra Vehicular Activities (EVAs) and another working within the shuttle itself. The use of spacecraft to move around and scan the shuttle exterior, either autonomously diagnosing potential problems or relaying all data to human operators for interpretation could drastically reduce the astronaut hours required. Small spacecraft would be able to gain access to very difficult to reach areas and in theory could not only be equipped with a variety of detection devices, but have the capability of carrying out repairs. An identical concept could be applied to the ISS for maintenance purposes. After the MIR space station accident, in which the Progress Craft impacted the station causing a breach in the hull and subsequent loss of pressure, it was necessary to use the Space Shuttle to observe the Spektr module for the breach location. This was a very hazardous and challenging mission for all per-

sonnel on both spacecraft: the same mission could have been carried out using an individual or fleet of smaller exploratory craft, without the high risk level.

The use of a 'space-rover' to look for potential problems has been widely investigated in the research community, although current focus has been based towards a single inspection vehicle. NASA has identified the use of robotic inspection of space vehicles as an important technology to reduce or even eliminate the need for EVAs. Current activity involves the continual development of the mini AERcam[24], though this research is focussed mainly on using a single entity rather than a group of such spacecraft completing synchronized imaging [44].

Formation of Large Structures As an extension to the desire of reducing the amount of EVA's completed by astronauts, it is conceivable to use a group of pico-satellites for the assembly of large structures from Sections inserted into orbit. Through dry-docking techniques, smaller 'slave satellites' could attach to and subsequently manipulate items. In a similar fashion to the collaborative action exhibited with ants in the carrying of larger food items, it is conceivable that larger items could be manipulated by using more of the 'slave satellites' to impart the required delta-V. Such methods could allow for the fabrication of very large structures without the need of any physical human presence. This implementation is somewhat ambitious considering the required delta-V production to form a significant space structure, though the ability of space 'mega-structures' to form autonomously has been investigated [50, 51, 52]. The core to such an implementation is the ability of each structural component to be autonomous and either reconfigurable or of modular design. By implementing techniques of self organisation, the structure can be formed autonomously without the need of human intervention.

Solar arrays and receiver arrays are perfectly scalable systems: solar arrays have a power output which is linearly dependant on the array size, and receiver arrays have a gain which is linearly dependant on effective aperture size. In addition, the elements forming the structure are largely homogenous. Consequently, both systems are suited to self assembly approaches: a huge benefit considering the challenge in assembling such massive structures in space. A recent ESA study explored the concept of forming a very large reflector array for power generation and subsequent transfer to Earth [52], and a preliminary study by NIAC has been made on the application of a sparse array for Earth Observation [53]. The construction of receiver arrays themselves present a new concept: the use of a phased array[25] to form a

[24] AERcam: Autonomous Extravehicular Robotic camera

[25] Phased arrays consist of multiple receiving elements dispersed over a large area to increase the possible

sparse aperture and remove the current mass limitation imposed by traditional filled apertures. The two studies illustrate perfectly how although agent solutions can be implemented to solve a multitude of technical issues, a multi-agent solution may be far from the optimal solution. It is undeniable that a multi-agent approach to the formation of large structures is of huge benefit and that formation of a sparse array represents the ideal arena for agent implementation since all receiver elements will be homogenous. However, within the implementation presented within [53], in addition to the need of a central collector, a companion satellite is required to act as an illuminator so that the desired level of resolution can be achieved from the swarm elements. Moreover, the resolution sought after is achievable through current monolithic platforms using current technology, indicating that a swarm approach is perhaps not the best option for such a mission [54].

Reconfigurable Spacecraft A modular robotic systems is defined within [55] as a robotic system constructed from a set of standardized components, or building blocks. Such robots are of interest, as they allow for the construction of a variety of specialized robots from standard components. An extension to such robotic systems exists within the field of reconfigurable robotics: at its most advanced a fully (self) reconfigurable robot would be able to use morphological properties to its advantage, becoming a snake like structure to move through a narrow enclosure, a legged robot to negotiate uneven terrain and even a wheeled shape to travel across a smooth surface with maximum efficiency. Although a highly ambitious example, it does illustrate the functional and economic advantages possible from a reconfigurable system. In terms of application, a modular reconfigurable system would be of great use for unknown and complex environments; space and planetary exploration present the extreme end of this unknown environment [56].

An overview of the current existing modular and potentially reconfigurable robots is given within [55] and [57]. It is not the intention here to replicate the work given within the papers but to identify that within the field of modular and reconfigurable robotics, despite the different hardware concepts, the basic building block requirements are the same:

- Each unit must be self contained and be capable of autonomous action. For this to be achievable, the unit must have at least some sensory information which can be processed locally, a power supply, means of communication and actuators.

resolution through an interferometric approach.

- The mechanism of connection between blocks should be genderless and, if possible, fail safe. In an ideal scenario, other blocks would be capable of removing faulty blocks from the system and disposing of them.
- Each unit should be small in dimensions. Although constrained by current technology, spatial dimension reduction is driven by the desire to gain access to confined spaces. Current systems produced by MEMS technology exist in the mm scale, though 'Claytronics' is a subset of this field which seeks to implement nano-scale robotics to form machines or mechanisms [58].

Application of reconfigurable robotic systems to space systems would yield numerous benefits and would address the current fundamental limitation that spacecraft are not mission flexible: they are designed for a specific mission and, once built, can not be applied effectively for any other purpose than was intended. Whilst some missions have served dual roles, such as the ISEE3 mission[26], the two assigned roles were not drastically different. What has not been achieved is the switch between drastically different functionalities, such as a telecommunications satellite taking the role of a weather satellite.

Introduction of reconfigurable modular robotics to space systems will not only open a new method of satellite development, which is expected to reduce the time and cost required for integration and ground test, but such satellite architectures will pose new problems for system design, information management and control system design. Of these factors, it is anticipated that the the control aspect will be the most challenging for successful implementation of a reconfigurable system [58].

Utilization of reconfigurable robotics in space systems is a concept closely linked to that of fractionated spacecraft, an idea being followed by both ESA and NASA to introduce flexibility, robustness and economy within their respective space programs [59]. The Panel Extension Satellite (PETSAT) concept, as presented within [59], proposes a satellite constructed of several functional panels each of which has a dedicated function such as a CPU panel, battery panel, communication panel or thruster panel. This concept has been taken a stage further wherein the functional blocks are not physically linked, but exist within a 'cloud' of components which communicate through a wireless network [60].

[26] The International Sun/Earth Explorer 3 (ISEE-3) satellite, launched August 12 1978, was part of the International Sun-Earth Explorer cooperative program between NASA and ESA to study the interaction between the Earth's magnetic field and the solar wind. ISEE 3 was the first libration point orbiting spacecraft, initially orbiting the Sun-Earth L_1 point before changing designation to ICE (International Cometary Explorer) and making the first visit to a comet, Giacobini-Zinner.

2.9 Summary of MAS and IMAS

The topic of multi-agent systems, intelligent or otherwise, encapsulates fields as disparate as biological systems, software and robotics. multi-agent robotic systems are an extension of multi-agent software systems: both of these systems aim to replicate the emergent behaviors observed within biological systems. In multi-agent robotic systems, a MAS is endowed with a body through which interaction with the environment is made possible. Despite being a relatively young area of research, the field is developing rapidly in all related areas: multi-agent software systems are already in use and multi-agent robotic systems are being used successfully within industry for non-life critical systems.

From a practical engineering standpoint, perhaps the most interesting multi-agent system abstraction is within the field of robotics. The suitability of MAS to highly complex and dynamic environments, combined with the desire to produce robust and economic systems, has led to the emergence of highly advanced concepts such as reconfigurable modular robotics: the pinnacle of swarm robotics. These complex robotic systems are themselves best controlled using an agent systems approach to maximize the potential of such systems.

Despite the reluctance to implement new technology without rigorous testing, the space sector has much to gain from the application of multi-agent systems concepts: the most significant gain can be achieved within the realms of exploration, fractionated spacecraft and the formation of large structures. The latter of these possibilities has been studied to some degree, concerning the formation of a solid structure and the replication of a phased array. Whilst the formation of a sparse sensing array for Earth observation was shown to have significant problems, the creation of an artifact which is intrinsically sparse in nature was not considered: the formation of a space based interferometer. Similar in concept to a phased array, it is potentially of more interest to observe the universe since a distributed array in space would allow for the detection of EM radiation normally absorbed by the Earth's atmosphere[27] and would not be affected by the increasing volume of air traffic.

2.10 Chapter Summary

This Chapter has given an introduction to multi-agent systems and how they have been applied within industry to date. The application of multi-agent systems in space, most

[27] Short wavelengths (gamma, X, UV and virtually all IR radiation) are subject to absorption in the lower atmosphere due to molecular oxygen and water vapour. At the opposite end of the spectrum, wavelengths greater than 10m are blocked as a result of the ionospheric F2 layer, which peaks at altitudes of 250-350km.

notably that of Remote Agent onboard the Deep Space One spacecraft, has been presented and the numerous alternative application possibilities for such systems within the space environment suggested. One of the presented application scenarios was that of deep space interferometry; the following Chapter will take this concept further, presenting the basic principles of interferometry, aperture synthesis and methods of applying these methods in a space environment.

3. INTERFEROMETRY

This Chapter presents the concept of interferometic imaging as a means of performing high resolution observations whilst circumventing the need for an unrealistically large single aperture device. Methods of aperture synthesis, wherein it is sought to achieve the resolution equivalent to that of a large single aperture device whilst minimizing the collection area of a sparse array, are introduced. Finally, the need for spacecraft interferometry is presented, in addition to current thoughts on how such a system may be realized.

3.1 Resolution Restrictions

The effectiveness of an image forming device is measured in terms of its angular resolution or minimum resolvable distance, which is the ability to discern between adjacent points in an object being viewed. There are various methods for determining the resolution of a monolithic device, either by use of spatial frequency or focal plane metrics[1]. Considering the latter, the maximum possible resolution of a traditional filled circular aperture is diffraction limited, being constrained by the physical size of the collecting aperture. Using diffraction theory, the 'Rayleigh Criterion' for the maximum resolution of a filled telescope, R_{filled}, with a diameter of D is given as:

$$R_{filled} = 1.22\frac{\lambda}{D} \text{ Radians} \tag{3.1}$$

which implies achieving high resolution observations entails the use of ever higher frequencies or the construction of larger diameter detectors. By example, considering the observation of a wavelength $\lambda = 1m$, to achieve a resolution of one arc second[2] would require an aperture over 250km in diameter, which is clearly not feasible technologically nor financially [61].

[1] In a strictly optical context, for a monolithic circular aperture, it is the width of the central peak within the point spread function which is of interest; this in turn relates directly to the aperture diameter

[2] An arc minute is a unit of angular measurement, equivalent to one sixtieth of a degree. An arc second is a further division of an arc minute: there are 60 arc seconds in an arc minute.

Interferometry is a technique which overcomes such limitations by using the information from numerous detectors resulting in a resolution dependant upon the detector separation, or baseline, rather than the physical size of a single detector. Considering this sparse configuration, where the apertures are separated by a baseline B, the resolution, R_{sparse}, is given by:

$$R_{sparse} = \frac{\lambda}{2B} \text{ Radians} \tag{3.2}$$

and so clearly the restriction of an impossibly large aperture is circumvented by using a collection of suitably spaced apertures [62]. Whilst it is true that the use of interferometers for high resolution observation does avoid the technological issue of constructing massive single apertures, it moves mechanical difficulties into issues of communications and data management.

The basic principles of interferometry are founded upon the wave properties of light as first observed and documented by Thomas Young in 1803. Modern implementations of interferometry adapt the familiar '2-Slit experiment' setup to the usage of EM-detectors and a beam combiner to replace the 'slits' and screen respectively. These spatially separated apertures transfer data to the combiner, also known as a correlator, where the radiation is interfered to produce image data. Within the context of stellar observation, the interference patterns can be used to study the brightness distribution of an object, as first proposed by Fizeau in 1868 and implemented in hardware by Michelson in 1891[3]. From these early beginnings, the use of interferometry for stellar observation has developed greatly; a complete documentation of the literature encompassing the theoretical and technological development within the field is far beyond the scope of this thesis and the interested reader should consider [63, 64, 65] for such details.

3.2 Interferometric Imaging

An image can be intuitively represented as a spatial intensity map; alike the operation of the human eye or a CCD[4]. In such an implementation, for each location in the spatial (x,y) plane an intensity level is given corresponding to the RGB levels or brightness. Within interferometry, a spacial frequency method is used in which it is the change between local spacial intensities that is of interest: the principal observables are the amplitude and phase of the complex visibility. A single interferometer (or a single baseline pair)

[3] Michelson and Pease first used an interferometric approach to measure the angular diameter of Jupiter's moons in 1891 and later to measure the diameter of Betelgeuse in 1920 [63]

[4] Charge-coupled device

evaluates the Fourier transform of the source brightness distribution for a particular value of the spatial frequency, given by the spatial frequency of the baseline vector, projected onto the u,v-plane [66]. The u,v-plane is the coordinate system used to represent the correlator response, which has coordinate distances expressed in wavelengths and related to the spatial x-y aperture locations by:

$$u = \frac{\pm(x_2 - x_1)}{\lambda} \tag{3.3}$$

$$v = \frac{\pm(y_2 - y_1)}{\lambda} \tag{3.4}$$

where (x_1, y_1) and (x_2, y_2) represent the spatial (x, y) locations of apertures 1 and 2 respectively.

Given 'enough' observations to develop the Fourier transform, techniques can be implemented to reconstruct the more familiar intensity image through inversion of the Fourier transform, as detailed within [66, 67, 68]. The quantification of 'enough' with respect to the number of baselines used to develop the Fourier transform is the subject of aperture synthesis.

3.2.1 Aperture Synthesis

Effective aperture synthesis seeks to achieve the maximum possible resolution, equivalent to that of a monolithic filled aperture of area A_{filled}, whilst minimizing the total collection area of the array, A_{sparse}. Consequently it is minimization of the fill factor, υ, which is sought. For an array of N identical sub-apertures of diameter D_{sparse}, the fill factor is given by:

$$\upsilon = \frac{\sum_{i=1}^{N} D_{sparse}^2}{D_{filled}^2}$$

The fill factor is not the only metric used to quantify the performance of a sparse aperture array; an equally important consideration is the obtainable signal to noise ratio which is inferred by the modulation transfer function (MTF) at the midrange spatial frequency and is the sparse analogue of the point spread function[5] for monolithic apertures [69]. Optical engineers have observed that this quantity is directly proportional to the fill factor:

[5] The point spread function describes the response of an imaging system to a point source and is the spreading of a point image over a finite area.

decreasing the fill factor reduces the midrange MTF, which in turn reduces the SNR and necessitates longer integration times [69, 70].

The original concept of aperture synthesis sought the attainment of complete u,v-plane filling so as to completely prescribe the Fourier transform; the essential link between interferometer observations and the brightness distribution. By generating a fully sampled Fourier transform, the information contained is identical to that produced from a single aperture telescope with a diameter equal to that of the maximum baseline length. Even if achievable, such extensive coverage of the u,v-plane is not required since it is possible to achieve useful imaging with a relatively sparse set of baselines. This is however at the expense of increased computation and integration time. A balance between the required computation and the number of apertures used to sample the u,v-plane is generally made and consequently the positioning of apertures is crucial to maximize their potential. Given N interferometer elements, there are $\frac{1}{2}(N^2 - N)$ unique baselines, each yielding a contribution to the Fourier function from which an inversion can be used to construct an estimate of the source brightness distribution. Due to the finite number of elements and hence finite number of baselines, it is of paramount importance not to replicate any measurements of the u,v-plane. This issue of asset management has received a large amount of attention regarding optimal element arrangements, enabling the maximum utility from the available resources [71, 72, 73, 61, 69, 74].

Both Golay and Cornwell considered a maximum configuration size of 12 elements applied to 'snapshot' imaging, within which short averaging times are implemented; the interferometer is used in a Fizeau interferometric mode alike the VLA[6] shown in Figure [3.1].

The initial research completed by Golay within [71] represents the limit of sparseness and yields a family of two-dimensional arrays with non-redundant and compact autocorrelation functions, meaning that the compactness of the u,v-plane filling is maximized without replication of u,v-point measurements. Within the work a family of patterns based upon square, hexagonal and triangular symmetries was derived. Although the methods used to do this were not detailed within the paper, the aperture coordinates are given within [69, 73] and reproduced within Appendix A.

Within [72], Cornwell develops a set of formations aimed at optimizing the uniformity of the u,v-plane coverage through simulated annealing techniques. In the application, the energy function was identified with the measure function $m(r_1, r_2, \ldots, r_N)$ as the element positions, here denoted by r_N, were varied. The measure function, within which self terms

[6] The Very Large Array is an interferometer array constructed in New Mexico and used for radio wave observations. It consists of 28 radio telescopes arranged in a 'Y' configuration yielding 351 independent baselines, with a maximum baseline length of 35km. The u,v-plane coverage is sufficient to allow implementation in a 'snapshot' mode [66]

Fig. 3.1: Aerial photograph of the VLA in New Mexico. Image courtesy of NRAO/AUI.

are ignored, is given by the set of difference vectors:

$$\begin{aligned} m(r_1, r_2, \ldots, r_N) &= \sum_{i,j,k,l} log\left(|u_{i,j} - u_{k,l}|\right) \\ u_{i,j} &= r_i - r_j \end{aligned}$$

The output produced a series of configurations based upon a circle: although the points were equidistant from a central location, they are not equidistant from each other. Due to their symmetric nature the arrays are redundant in rotation and so are of greatest use in correlation arrays, for which the instantaneous coverage must be very good. Within the paper, Cornwell highlights the problem of implementing such non-redundant arrays due to *'the sensitivity to temporarily missing or nonfunctional elements'*. Such concerns are typical of sparse systems where issues of robust operation are of paramount concern, in this instance the ability to compensate for faulty elements is made apparent.

3.3 Spacecraft Interferometry

Numerous interferometry facilities have be constructed since the Mt. Wilson 100 inch telescope used by Michelson and Pease, allowing observation of different wavelengths

with ever increasing resolution. A catalogue of the facilities through the ages, their respective sizes, target wavelength and observational output is given within [63]. Despite the construction differences within these facilities, they do have a single commonality: they are all Earth based. Earth based observation of stellar sources, although convenient, is non ideal as atmospheric interference results in the problems of distortion and absorption:

Distortion The atmosphere is turbulent and results in distortion in the wavefront due to varying densities causing the effective pathlength to vary. Should these introduced distortions become a significant fraction of the wavelength then the resolution is no longer dependant on the size of the primary aperture, but the coherence of the incoming wave front [66].

Absorption Short wavelengths (gamma, X, UV and virtually all IR radiation) are subject to absorption in the lower atmosphere due to molecular oxygen and water vapour. At the opposite end of the spectrum, wavelengths greater than 10m are blocked as a result of the ionospheric F2 layer, which peaks at altitudes of 250-350km.

Using current technology, atmospheric distortion can be compensated for on a ground environment through use of adaptive optics, as implemented on the VLTI[7], Keck-I and II[8] and CHARA Array[9] [63]. Although addressing the problem of atmospheric distortion, the issues of atmospheric absorption remain and so observation of a significant proportion of wavelengths is only possible above the atmosphere: this directly implicates the necessity of space based optics. This need for space based hardware to permit clear observation is well established and exemplified with the current Hubble Space Telescope (HST), a single aperture optical device which is diffraction limited to a maximum resolution of 0.1 arc-seconds. This maximum resolution was set by the maximum permissable diameter of the primary mirror, which itself was limited by launch vehicle constraints. Such a constraint reiterates the aforementioned physical size restrictions which single aperture configurations suffer; a constraint which is far more severe when applied to a space application. Methods in circumventing such volume constraints have been sought and the use of inflatable structure technology has been proposed for the construction of large aperture antennae and reflectors due to their inherent low mass and volume [73, 75, 76, 77].

[7] The Very Large Telescope Interferometer consists of 8 telescopes (4 primary, augmented by 4 moveable) and is located at the Paranal Observatory on Cerro Paranal, a 2,635 m high mountain in the Atacama desert in northern Chile.

[8] Keck-I and II are the two optical telescopes which constitute the NASA-funded interferometric observatory at Mauna Kea, Hawaii.

[9] The Center for High Angular Resolution Astronomy (CHARA) is an optical interferometric array of six telescopes located on Mount Wilson, California

Despite these appealing qualities, the application of such technology is still abound with problems such as attaining sufficient rigidity and radiation induced degradation. In addition, the fact remains that at some point even an inflatable structure will have a size constraint. The only foreseeable method to circumvent the restrictions imposed upon a space based imaging system from either launch constraints or technological limitations is to use a sparse aperture physically separated spacecraft interferometry system (PSSI)[10]. There are two primary methods for implementing such a system: a tethered system or free flying elements.

Tethered A tethered system involves a central combiner to which elements are attached via tethers [78]. The entire system rotates alike a wheel, with the combiner at the center and the elements maintaining a fixed location with respect to the hub center. Such a system is advantageous in that as the elements rotate and occupy new locations in the physical x,y-plane, new u,v-plane measurements can be made. This scanning operation mode allows for greater filling of the u,v-plane with minimal usage of propellant and fewer apertures . In addition to efficient implementation of a scanning mode, the tethers can also be used as a communication line to the central combiner. Despite these advantages, a tethered system is limited in other ways: baseline changes are limited to the tether length and reconfigurability/replacability in the instance of element failure is severely limited. In addition, a tethered system requires launch as a complete system and consequently launch constraints are again a consideration.

Free-flying A free-flying system, as investigated by ESA and NASA, involves physically detached elements maintaining relative positions through propulsive means. Whilst this allows for greater flexibility in element reconfigurability or replacement, it is at the expense of increased fuel usage and complexity of all systems associated with data handling and state determination. Methods have been sought to allow implementation of a free flying system in a fuel efficient manner by investigating beneficial gravitational environments such as those which exist in libration point orbits [79].

The concept of such interferometric systems is not new: DARWIN and TPF-1 are the respective programs from ESA and NASA implementing such techniques. The two programs are essentially identical, with the aim of detecting Earth like planets through characterization of planet composition. The intention is to use free-flying platforms located at

[10] Within the current literature, it is common to use the acronym SSI for 'separated spacecraft interferometry'. However since the separation is only within the kinetics and not communications or general operation, the modified acronym of PSSI shall be used and retained throughout this thesis.

the Sun-Earth/Moon L_2 Lagrange point to implement a nulling interferometer[11] operating in the Infra-red spectrum[12] in combination with spectroscopy [81, 82, 83]. Despite the global interest in such high profile missions and the full weight of ESA and NASA scientists designing the respective missions, implementation is still far from reality; to date although potential flying configurations have been investigated for the DARWIN mission within [81], the formation to fly has not been decided upon and the hardware required to perform the desired tasks has not been constructed, let alone performed the pre-requisite space qualification tests.

The requirements of a PSSI system are demanding both due to the nature of the science which is to be completed and the environment within which the science payload is to operate. The Darwin and TPF missions require autonomous behaviour of the individual elements and the system as whole. The need for autonomy in the system is driven by the time delays involved in communication at such large distances from Earth, in combination with the strict formation flight requirements. Individually each element must be precisely regulated to a relative state in six degrees of freedom, known to within fractions of the wavelength being observed. As a group the entire system must be capable of attaining a specified configuration and orientation in the celestial sphere. Additional system requirements arise with the need to provide fault tolerance should one or more of the interferometer elements fail either partially or fully.

3.3.1 PSSI as a Multi-Agent System

Whilst research has been completed on PSSI within Lagrangian and Earth orbiting dynamics, current work has been firmly in the traditional context of satellite formation flight and not that of multi-agent systems. PSSI can clearly be considered as a multi-agent system, with each interferometer element an autonomous agent who's task is to maximize the utility of the overall configuration. The application of these principles within the context of PSSI allows for the direct transfer of all the multi-agent systems advantages put forth within Chapter 2, allowing for a considerably advantageous control architecture.

The inherent autonomy aspect provided by multi-agent systems can allow for the satisfaction of the anticipated PSSI requirements regarding self regulation. In addition, with the ethos of a multi-agent system being that of beneficial group behavior, the desire to maximize scientific output in the presence of individual agent failure is naturally addressed.

[11] Nulling interferometry is used for planet search operations as it is required to 'null' the output from the star which is being orbited by deconstructive interference of the on-axis star signal [80].

[12] The envisaged biomarkers of water, ozone, methane or carbon-dioxide determine the operational wavelength range in the mid-infrared from 6.5 to 20μm wavelength [81]

3.4 Chapter Summary

This Chapter has provided a brief introduction to the principles of interferometry and the rationale behind implementing space based interferometric imaging through either a tethered or free-flying system. Current insight on a space based interferometer is that of a distributed system, though it is possibly best described as a natural physical multi-agent system. The following Chapter will consider control and decision methods applicable to such a system.

4. PSSI CONTROL LITERATURE

The insight of PSSI as a multiagent system brings together the research areas of spacecraft formation flight (SFF) and multi-agent systems, itself linked to swarm stability. Whilst there is a wealth of literature regarding single satellite regulation to a nominal halo orbit, following from the initial treatment of control and use of libration point satellites within [84], this spectrum of work will be omitted from review since it is the formation flying aspect which is the focus of this thesis. SFF has been highly studied at the University of Southampton, though interest has been primarily centered around Earth orbiting spacecraft [85, 86, 87, 88, 89, 90, 91, 92]. Indeed, the subject of formation flying for Earth orbiting spacecraft has been highly researched by numerous institutions and authors since the birth of the Apollo program, the list of which is too great to be printed here. Within this thesis it is SFF within the dynamical regime of the CRTBP which is of interest, in particular SFF within Halo orbits, and this topic will now be presented.

4.1 SFF in Halo Orbits

There are currently two main research directions towards the solution of satellite motion in halo orbits: those from a strictly mathematical perspective and those based upon more traditional control solutions. The former, presented by Howell, Gomez and Scheers have published results primarily focussed on the mathematics of halo orbits in order to take maximum advantage of the dynamic environment [93, 94, 95, 96, 97, 98]. The later, although obviously becoming involved in the mathematics associated with Lagrange points, follow the more familiar feedback control regimes for self regulation as has been applied to single satellite orbit regulation [84, 99].

Traditional halo orbit SFF control solutions have predominantly concerned a leader-follower configuration within which only the follower is controlled, as presented within [100, 101]. Within [93] a halo orbit expressed in the Hill problem is investigated and a method of producing winding motions about a stable halo trajectory formulated. Such winding motions, revisited within [94] where problems of stabilizing such a free flying formation are given, are of relevance to interferometer formations implementing aper-

ture rotation to maximize u,v-plane coverage. Whilst such results are promising in minimization of fuel consumption, these problems would be avoided altogether for a rotating platform through use of a tethered formation [78]. Initial station keeping solutions provided by Howell and Gomez involve a target point strategy and a Floquet Mode approach, which use maneuvers executed impulsively at discrete time intervals [95]. These positional control methods are extended to multiple satellites within [96]. More complete solutions are provided within [97, 98], within which distributed solutions are developed. [97] uses a linear quadratic regulator (LQR) and feedback linearization methods applied to two elements: one stabilized to an unstable orbit, the other regulated to a non-natural orbit relative to the leader. This construct is similar to that presented within [101], who uses adaptive output feedback control to remove the need for velocity sensors. Although the control method developed within [97] is theoretically extendable to multiple elements and that of [101] to a leader follower configuration of the n^{th} order, this is not completed and the subject of collision avoidance is not considered in either implementation. [98] provides what is claimed to be a decentralized approach to allow distribution to a specified formation. Whilst [98] is classifiable as distributed, since the optimal trajectories are computed through multiple processors, it is not the elements themselves completing the computation and so the method still clings to a centralized method with regards to the agent community.

All of the papers published utilize continuous time methods within the control implementation, indeed those claiming a discrete time implementation involve control action based upon continuous time observables implemented at low frequency discrete time instances [95]. More significantly, the vast majority of papers relating to the subject of formation flight are claimed to be autonomous, when in actuality the term 'automated' would be more appropriate; within implementation control methods are applied without any real form of reasoning behind application of any routines. The closest paper implementing any form of reasoning behind the application of control is that of [98] which considers dispersal to a specified configuration; however this was neither truly decentralized nor resulting in an optimal configuration. The configuration adopted was that of equidistant apertures around the circumference of a circle, with the center on a nominal halo orbit; equidistant aperture locations result in replication of u,v-plane measurements and represents a mismanagement of resources. Whilst reactive behavioral methods have been applied to the space environment within [102, 103, 52], such methods still not do apply any reasoning within implementation and consequently may still be considered as automated; these behavioral methods will be revisited in a subsequent Section.

Within all the referenced papers, precise knowledge of the spacecraft state is assumed when in reality such accurate knowledge is impossible to attain: this is particularly pertinent to those seeking advantages in the dynamic environment. For past libration point

missions, the Deep Space Network (DSN) has provided tracking services with accuracies varying between 2 and 10km for MAP and ACE respectively [104]. Due to the increasing burden on the DSN, this has become a less viable option and so research efforts have been made in developing an autonomous method of celestial navigation, culminating in CelNav which has a maximum accuracy of 9km and 4mm/s [105]. Whilst such errors are acceptable for orbital maintenance of a single satellite, interferometry requires accurate knowledge of each element location to within fractions of the wavelength being observed; a requirement far beyond the capabilities of CelNav and directly implicating the need for high accuracy *local* distance measurement devices. Although the respective space agencies directly specify this requirement and some research papers hint at the need for obtaining accurate local state measurements, no papers integrate methods on the development and utilization of a suitable coordinate system resultant from the implementation of realistic sensor measurements.

At present, methods to maintain optimal interferometer performance upon element failure or to prolong mission lifetime, have not been considered. This is perhaps the greatest advantage to implementing PSSI as a multi-agent system since such considerations are an inherent trait from the desire to maximize system utility.

4.2 MAS within PSSI

Incorporating the autonomous control exhibited by formation flying routines with agent systems principles seeks to increase the possible autonomy from such systems to include aspects such as pro-activity, learning and adaption [106, 107, 108]. Following from the introduction given within Chapter 2, an agent can be described concisely as a pro-active object. In the context of PSSI, the pro-active nature of an agent seeks to form a suitable sparse aperture array which represents the maximum possible utility of the agent community. The ability of each agent to acquire and subsequently maintain a particular position is the subject of SFF, which has already been addressed. Whilst Chapter 2 has presented an overview of multi-agent systems, implementation of PSSI as a multi-agent system is perhaps best abstracted in the context of robotic systems rather than that exhibited by the Remote Agent experiment: what is of interest here is the decision processes involved within the agent community to allocate optimal agent locations[1] and the internal dynamics of the system in the presence of agent failure.

For the instance of optimal aperture formation, it is entirely possible for the agents to autonomously investigate and achieve an optimal formation; in essence physically conducting the simulated annealing techniques implemented within [72] and the ant opti-

[1] Here optimal agent locations correspond to maximizing the u,v-plane coverage.

mization techniques implemented within [20, 24]. Although possible, such a methodology does not represent a practical implementation, since excessive amounts of propellant would be consumed during the investigative stage. For PSSI implementation, it is not required to search for an optimal configuration as this has already been achieved [71, 72]. The problem at hand now is how to achieve such configurations within a group of self-interested agents: coordination is required so that all the locations of a given set are reached by an agent, but are otherwise all agents are independent.

4.2.1 Agent Action By Behavior

Behavioral approaches to MAS actions have been investigated by numerous authors: such methods seek to evolve desirable system traits through reactive behaviors, as observed by biological systems. [109] sought to achieve desirable emergent traits by prescribing a set primitive behaviors and synthesizing more complex behaviors through combinations of these primitives. Within their work, which was extended to application on mobile robots, they highlight the issues involved in avoiding undesirable higher level behaviors; this problem of achieving dependability is examined within [110]. Behavioral control approaches applied to ground based robotics are also investigated within [111], within which geometric positions are specified and maintained[2].

Behavioral methods are applied to the space environment within [102, 103, 52]. These papers use potential functions to prescribe desired responses which are enforced through sliding mode control. [52] extends the work within [102, 103] and in a similar manner to [109], combines various weighted potential functions (the primitives) to result in higher level behaviors. This methodology was termed as 'equilibrium shaping'. Potential functions have been used greatly within the realm of autonomous robots for purposes of navigation, but are fraught with problems of local minima and the generation of suboptimal trajectories.

The reactive nature of behavioral control methods are highly suited to very large scale systems which require autonomous configuration into simple, regular structures. Employing the 'equilibrium shaping' mechanism from [52] with potential functions introducing avoidance, sphere formation and plane formation behaviors, would allow for the aggregation to an equidistant circular array identical to that within [98], which has already been rebuked as not being optimal. Whilst it is conceivable to alter the individual behavioral weights on an agent-by-agent basis to more closely emulate a non-regular Cornwell array, we immediately enter the problems associated with formal verification of behavioral methods. In particular the dependability of the system to repeatedly achieve the desired

[2] Note that a global configuration is not specified; only interagent distances and directions are sought.

formation would be a major consideration [110].

4.2.2 Agent Action By Decision

Within [1], numerous mechanisms for agent decision making processes are presented, all of which fall under the umbrella of competitive market analysis. In the context of an PSSI system, there are two methods of achieving the desired agent action: centralized planning with distributed action (CPDA) or distributed planning and action (DPA). CPDA involves a centralized decision process with the output of this decision being implemented by a distributed set of agents, which are themselves capable of completing the required task without supervision. A DPA scheme involves decisions being made and executed by the agents without any form of supervision at any stage; once an agreement has been reached by the agents involved, the agreed action is then executed. Whilst behavioral methods could be considered as a weak form of DPA, they are not considered here due to the lack of inter-agent communication during the decision process and their reactive nature.

Visualization of CPDA within PSSI is clear: in this instance a central-satellite would be responsible for formulating decisions and prescribing agent action, based upon the desired interferometric arrangement and the current state of the perceived world environment. Once an agent has been assigned a location it autonomously moves to and maintains the prescribed position using the relevant skills at it's disposal. Implementation of DPA requires that all agents communicate and decide within their community which agent should move to which position. Once an agreement has been formulated, the agents move to and maintain the agreed positions. Both methods seek to achieve the same result: efficient allocation of resources to maximize the u,v-plane configuration utility for interferometry. In the case of centralized planning the task is to globally minimize a cost function for all the agent members; for distributed planning the task is to reach an agreement between self-interested agents each seeking to minimize their own cost function. Although different in implementation, the cost function which is to be minimized is the same for both methodologies and relates to motion of agents to specific aperture coordinates. Both CPDA and DPA schemes have their associated benefits and disadvantages: CPDA methods can easily produce a globally minimum solution at the expense of computation and for DPA methods the converse is true. A hybrid auction format would combine the benefits of both methods by using each agent to calculate their own cost function, then transferring this information to a central agent which can irrefutably assign an optimal allocation, without the need of a lengthy agent communications within the decision process.

The most popular implementation for prescribed agent decision processes is that of an auction system, which is an intrinsically decentralized method and presented within

[112, 113, 114, 115]. [112] reviews and details auction methods to coordinate terrestrial robots for accomplishing tasks such as exploration [113, 114] and task allocation [115]. Whilst auction methods are a popular decision method for MAS, there are no instances of such implementations being completed for a space based multi-agent system and as yet the heuristics remain unexplored. An agent auction process applied to PSSI would allow for optimal agent action to form a synthetic aperture and yield the possibility of doing so in a computationally efficient manner, within which processing is distributed throughout the agent community. It is the opinion of this author that such an implementation represents the optimal use of agent resources with the space domain.

4.2.3 Optimal Performance Under Failures

In the presence of agent failure, it is desired to maximize the utility of the remaining agents: this entails autonomous reconfiguration of the remaining agents into a new optimal array.

Upon departure of the failed agent from the community, behavioral approaches would permit autonomous redistribution of the remaining agents; but again we are presented with the problems of guaranteeing repeatability and the probability of settling into a non-optimal formation. Implementation of a formal decision provides a much more efficient and verifiable redistribution mechanism: upon detection of a failed agent, a decision relating to the optimal reallocation of agent aperture locations can be made. This decision involves determining the most efficient method for the N functional agents to move from their current aperture location assignment, which relates to an optimal $(N+1)$ array, to an *a priori* known optimal array of N agents. Once the formal decision has been made, the functioning agents move to their newly assigned array point locations.

4.3 Chapter Summary

This Chapter has presented literature associated with halo orbit control methods and mechanisms for agent decision making through behavioral and formal processes. It was highlighted that the majority of control methods rely upon continuous time methods focused upon a leader-follower configuration; more complete solutions have surfaced though none providing a complete solution. It was also highlighted that the control methods for solution to the formation flight problem are generally described as autonomous within the literature, whilst automated would be a more descriptive term, since the simulated entities are not making any real 'decisions'. Formal decision methods have been presented as superior to those of behavioral methods, due to their absolute nature and

methods to apply formal decisions within the multi-agent interferometer scenario have been suggested.

The following Chapter will serve to present the envisaged physically separated spacecraft interferometry scenario that is to be investigated, indicating the mission phases and instances of decision making.

5. THE ENVISAGED PSSI SYSTEM

This thesis will consider physically separated spacecraft interferometry (PSSI) as a multi-agent system and address issues of multi-agent ejection from a central spacecraft, subsequent decentralized and discrete time control to acquire an optimal interferometric formation about the central spacecraft and agent decision processes for optimal initial distribution and redistribution in the presence of agent failure.

Prior to entering upon material relating to the development and simulation of the PSSI system, an introduction to the intended PSSI scenario will be given for clarification to the reader.

5.1 PSSI System Operation

The envisaged operation of the multi-agent PSSI system is based upon a group of ten holonomic interferometer agents, as depicted within Figure 5.1, operating collectively to perform interferometry. It is assumed that the operational orbit for the agent system is reached by a 'mother satellite' within which the agents are stowed in a deactivated state during transport[1], as depicted within Figure 5.2. The operational environment for the application is that of a halo orbit, a class of libration point orbit. A libration point is a location within the three-body problem (in this case the Sun, Earth/Moon and a spacecraft) where the gravitational and centrifugal forces acting on the spacecraft balance; once stationed at this point a infinitesimal mass could theoretically remain there indefinitely. Orbits about these points fall into two categories: halo orbits are closed trajectories about these points, whilst Lissajous trajectories encompass bounded but non-repeating orbit tracks about libration points. Such locations are favorable due to the benefit in the gravitational environment over that of an Earth orbit: it is for this reason that a libration point orbit has been selected.

Once the mother satellite has reached the intended operational orbit, the interferometer

[1] The agents are assumed to be dormant during their transportation to the mission orbit; whilst it is entirely possible that the agents themselves could be used to distribute computation during to the transfer the operating orbit, this is a technical issue and not considered here.

agents are activated and disperse from the central craft to form an optimal Golay array, as presented within Chapter 3, wherein each agent acquires an array-point from the optimal set. The initiation of this process and the final configuration of the assembled array (in this instance a Golay-10 array) are depicted within Figures 5.3 and 5.4 respectively. A Golay array was selected since such configurations represent optimized formations for low baseline counts.

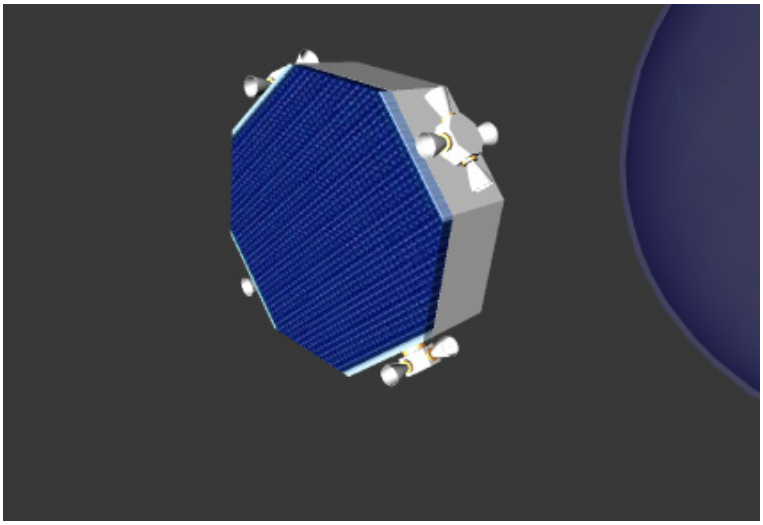


Fig. 5.1: Artistic representation of a single interferometer agent which is holonomic in control. Note that no engineering design has gone into the production of this model: it is for representation only.

Chapter 4 has already presented that the optimal aperture locations for an interferometric configuration, given a known number of apertures, may be determined *a priori* to implementation: it is the decision regarding agent location allocation which is the pertinent consideration upon initial agent dispersal. Upon system initialization it is assumed that the number of functional agents[2] are known, as is the corresponding optimal configuration for this number of agents. Each agent is to be assigned an array point location, for brevity this shall be denoted as the agent array point (AAP). The initial AAP allocations are not assumed available and this is the first agent decision to be made.

Upon completion of the decision process relating to the array-point allocation, each agent is tasked with obtaining and maintaining their allocated array-point, thus resulting in the desired configuration as depicted within Figure 5.4. Since the central mother craft

[2] Here the terminology of aperture and agent is synonymous

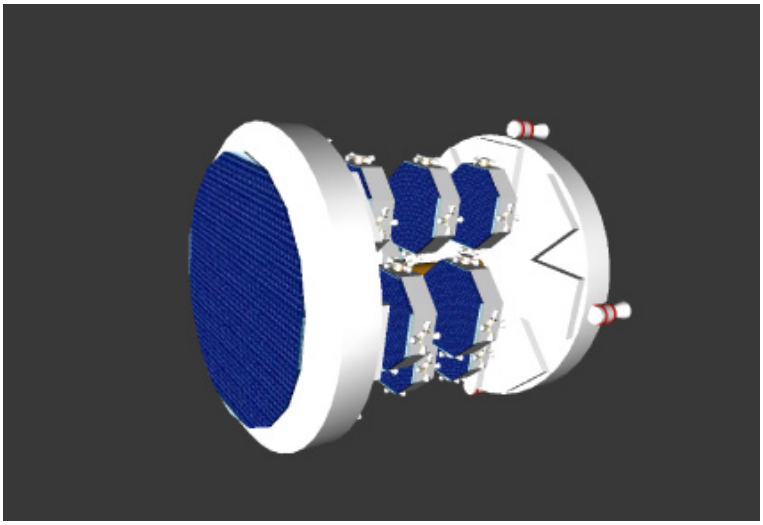


Fig. 5.2: Conceptual representation of the interferometer multi-agent system in a stowed configuration within a 'mother-satellite' prior to dispersal and implementation. Note that no engineering design has gone into the production of this model: it is for representation only.

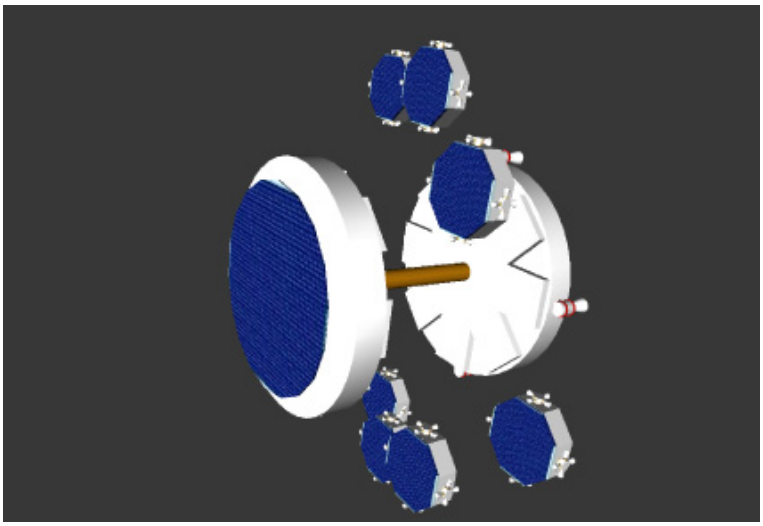


Fig. 5.3: Artistic representation of interferometric agent dispersal. Note that no engineering design has gone into the production of this model: it is for representation only.

is restricted to a halo orbit and is non-stationary, each agent will be required to perform

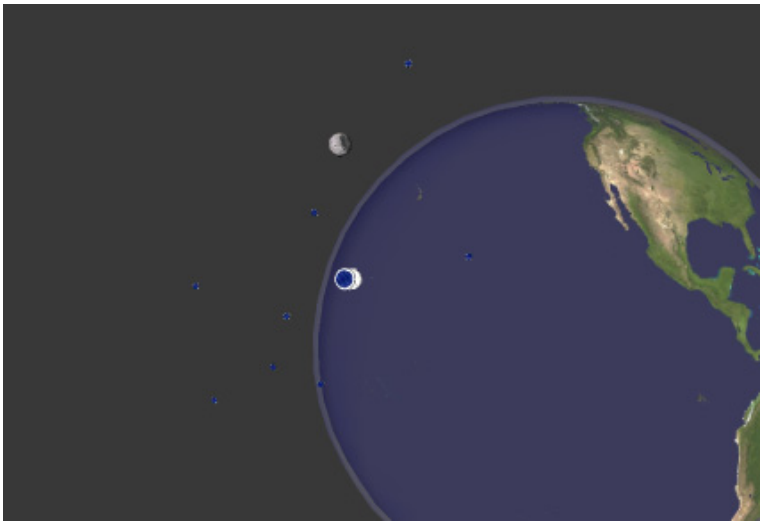


Fig. 5.4: Artistic representation of final interferometric configuration and main operational mode, having formed a Golay-10 array. Note that no engineering design has gone into the production of this model: it is for representation only.

position regulation in combination with translational velocity tracking in order to retain rigidity within the array. In addition to this, each agent must be capable of regulating their own attitude and attitude rate to enable pointing and tracking of a target source within the celestial sphere.

This represents the main operational stage of the mission: each agent receives and transfers data to a central craft (nominally the mother satellite) which is used to combine the receiver data into a coherent data entity for subsequent transmission to an Earth based ground station. This interferometric process has been presented within Chapter 3.

During the mission lifetime each agent will remain informed upon the health of surrounding agents: should an agent fail then a new optimal arrangement, relating to the number of remaining agents, must be obtained by a formal decision method, the results of which are executed autonomously by the agent community. An additional requirement for the agent system is to maximize the mission lifetime; this requirement is to be satisfied through sequenced placement of the agent resources such that no single agent is continually subject to excessive fuel demand.

5.2 Chapter Summary & Thesis Direction

This Chapter has presented the scenario which is to be investigated: that of multiple spacecraft agents dispersing from a centralized location, forming a decision relating to array point allocation and subsequent control regulating the agents to the specific locations decided upon.

The remainder of this thesis will concentrate upon the required aspects to create a faithful simulation of the agent system, comprising of dynamical simulation, decision processes and control action.

Part II will present the dynamics of the circular restricted three body problem (CRTBP), six degree of freedom dynamics and kinematics for a holonomic satellite and guidance methodologies. The presented guidance and dynamics will be used within subsequent Sections to develop a control methodology and permit computer simulation.

Part II

MATHEMATICAL MODELING

6. THE CIRCULAR RESTRICTED THREE BODY PROBLEM

The circular restricted three body problem (CRTBP), as formulated by Euler in 1772 for the Sun-Earth-Moon system to study the perturbed motion of the moon about the Earth, is a mathematically degenerate case of the general three-body problem which dates back to Newton's investigations in the 17^{th} Century. The problem is defined within [79] as: *two bodies revolve around their center of mass in circular orbits under the influence of their mutual gravitational attraction and a third body (attracted by the previous two but not influencing their motion) moves in the plane defined by the two revolving bodies. The restricted problem of three bodies is to describe the motion of this third body*. An extension of this case is the elliptic restricted three-body problem (ERTBP), within which the distance of the two primary bodies is permitted to vary periodically.

A large amount of literature regarding both the CRTBP and ERTBP exists, primarily originating from Poincaré's work presented within the three volumes of Méthodes Nouvelles completed in 1899. Szebehely's treatise [79], published in 1967, provided a comprehensive summary of the research concerning the restricted three body problem at the time. Interest emerged in the existence of three dimensional bounded trajectories, which although non-integrable, were pursued analytically by Richardson [116] and numerically by Farquhar [84] and Howell [94, 117], with the former author being concerned with practical applications.

Although not providing a comprehensive treatment of the CRTBP and all related issues, this Chapter will present all material pertinent to the subject of this thesis. Following a mathematical development of the CRTBP[1] and localization of the so called 'libration points', the dynamics about these points will be investigated, leading to the existence of both Lissajous and halo orbits. It is these trajectories which have become of great interest to the space science community and are of direct relevance to this thesis.

[1] The development here will be aimed primarily at the Sun-Earth/Moon system though equally valid for any valid three body system

6.1 CRTBP Formulation

A schematic layout of the synodic CRTBP system is given within Figure [6.1], where M_1 and M_2 represent the bodies of significant mass (primaries), at distance D apart along the X-axis and rotating with constant angular velocity n about their barycenter at distances D_1 and D_2 respectively. The system barycenter is the origin of the Z-axis and the system rotates about this axis on the Y-plane; it is this Y-axis that completes the rotating orthogonal coordinate system.

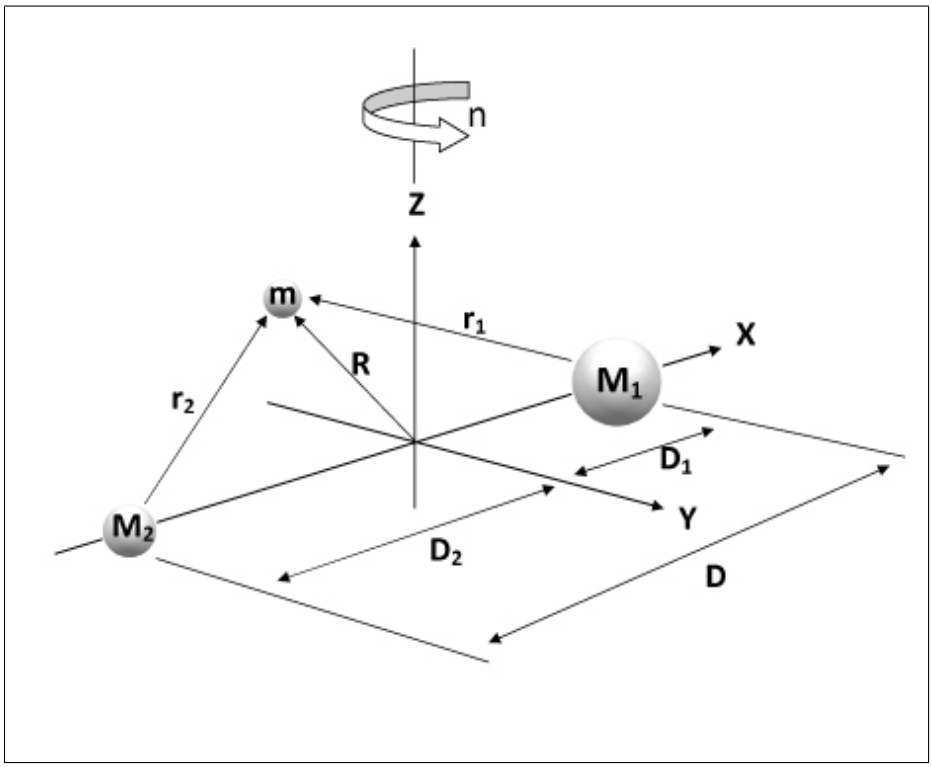


Fig. 6.1: Schematic layout of the CRTBP

For the scenario considered within this thesis, M_1 represents the Sun and M_2 represents the Earth/Moon system. The infinitesimal mass, which moves within the gravitational potential of the primary masses, is denoted by m and for the complete system $M_1 > M_2 \gg m$. Vector $\mathbf{R}$ is the distance from the system barycenter to the infinitesimal mass which can be considered as a spacecraft agent, vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ are the distances from the primaries to the agent. It is shown within [99] that the non-dimensional equations of motion, where distances are in units of D and time in units of $\frac{1}{n}$, are given by:

$$\ddot{X} - 2\dot{Y} - X = -\frac{(1-\rho)(X-\rho)}{\mathbf{r}_1^3} - \frac{\rho(X+1-\rho)}{\mathbf{r}_2^3} \tag{6.1}$$

$$\ddot{Y} + 2\dot{X} - Y = -\frac{(1-\rho)Y}{\mathbf{r}_1^3} - \frac{Y\rho}{\mathbf{r}_2^3} \tag{6.2}$$

$$\ddot{Z} = -\frac{(1-\rho)Z}{\mathbf{r}_1^3} - \frac{Z\rho}{\mathbf{r}_2^3} \tag{6.3}$$

where G represents the universal gravitational constant and

$$\mathbf{r}_1 = \sqrt{(X-\rho)^2 + Y^2 + Z^2} \tag{6.4}$$

$$\mathbf{r}_2 = \sqrt{(X+1-\rho)^2 + Y^2 + Z^2} \tag{6.5}$$

$$n = \sqrt{\frac{G(M_1+M_2)}{D^3}} \tag{6.6}$$

$$\rho = \frac{M_2}{M_1+M_2} \tag{6.7}$$

A more compact form can be produced by introducing a pseudo-potential function, U, defined as:

$$U = \frac{1}{2}(X^2+Y^2) + \frac{1-\rho}{\mathbf{r}_1} + \frac{\rho}{\mathbf{r}_2}$$

resulting in the three second order differential equations being:

$$\ddot{X} - 2\dot{Y} = \frac{\partial U}{\partial X} \tag{6.8}$$

$$\ddot{Y} + 2\dot{X} = \frac{\partial U}{\partial Y} \tag{6.9}$$

$$\ddot{Z} = \frac{\partial U}{\partial Z} \tag{6.10}$$

By defining a state vector as $\mathbf{X} = [X, Y, Z, \dot{X}, \dot{Y}, \dot{Z}]^T$, these equations of motion can be written concisely in state-space form as:

$$\dot{\mathbf{X}} = \mathbf{A}\mathbf{X} \tag{6.11}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ U_{xx} & U_{xy} & U_{xz} & 0 & 2 & 0 \\ U_{yx} & U_{yy} & U_{yz} & -2 & 0 & 0 \\ U_{zx} & U_{zy} & U_{zz} & 0 & 0 & 0 \end{bmatrix}$$

for which the second partial derivatives of the three-body pseudo-potential are given within Appendix B.

6.2 Libration Points

Libration points are the five equilibrium solutions to the equations of motion for the CRTBP, as shown within Figure 6.2. This yields the locations at which the gravitational and centrifugal forces acting on a third body in the synodic frame are balanced. A body at one of these locations will, when viewed in the synodic frame, remain stationary. Whilst Euler identified the three collinear libration points ($L_1 - L_3$) in 1762, it was Lagrange who identified the two triangular libration points (L_4, L_5) in 1772; libration points are referred to interchangeably as stationary points, libration points or Lagrange points. For the remainder of this thesis, only the term libration points will be used.

As has been mentioned there is a set of five libration points, formed of two subsets: the collinear and equilateral points. The set of three collinear points lie along the X-axis which joins the two primary bodies. One point exists internally to the primaries and the remaining two points are external to the primaries: one each on the far side of the primaries with respect to the system barycenter. The set of two triangular libration points are each positioned at the apex of an equilateral triangle formed with the primaries. The most common notation for the libration points defines the set of collinear points as L_1, L_2 and L_3 with the interior libration point as L_1, the point on the far side of M_1 as L_1 and that beyond M_2 as L_2. The triangular libration points are denoted as L_4 and L_5, with L_4 being the point moving in advance of L_5 when considering the synodic frame.

6.2.1 Determination of the Libration Point Locations

The libration points are found by setting the derivatives of equations (6.1) to (6.3) to zero[2], resulting in the following equations:

[2] Or equivalently those of the pseudo-potential function, U, to zero.

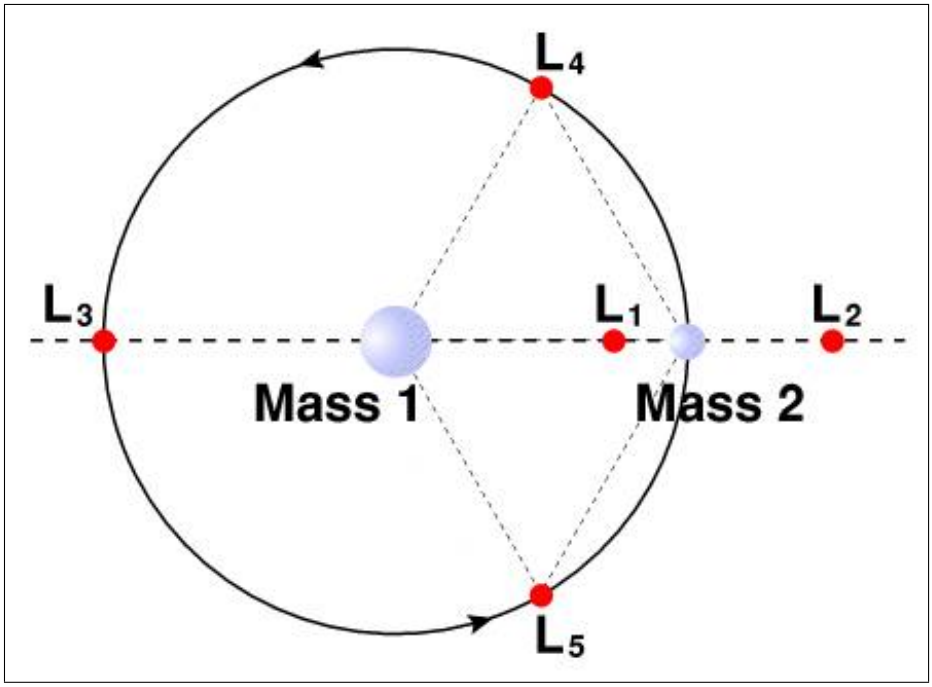


Fig. 6.2: The five libration point locations for the CRTBP. Image curtesy of the NASA Apollo 15 Flight Journal.

$$X - \frac{(1-\rho)(X-\rho)}{r_1^3} - \frac{\rho(X+1-\rho)}{r_2^3} = 0 \tag{6.12}$$

$$Y\left(1 - \frac{(1-\rho)}{r_1^3} - \frac{\rho}{r_2^3}\right) = 0 \tag{6.13}$$

$$\frac{(1-\rho)Z}{\mathbf{r}_1^3} + \frac{Z\rho}{\mathbf{r}_2^3} = 0 \tag{6.14}$$

The simplest libration points to locate are the equilateral libration points, L_4 and L_5. A solution to Equations (6.12) and (6.13) exists for when $Y \neq 0$ and $r_1 = r_2$. Imposing this condition and setting (6.5) and (6.6) to be equal results in $X = \rho - \frac{1}{2}$ and upon substitution of this value back into 6.5 we yield the Y location, which is at $Y = \pm\frac{\sqrt{3}}{2}$. Here the $\pm$ denotes the difference between the L_4 and L_5 points.

Determination of the co-linear libration points for the CRTBP is a somewhat more involved than that for the equilateral libration points. In the determination of the co-linear points, we are considering the case when both Y and Z are zero: imposing this condition onto (6.9) results in the 5^{th} order algebraic equation

$$X \pm \frac{(1-\rho)}{(X-\rho)^2} \pm \frac{\rho}{(X+1-\rho)^2} = 0 \tag{6.15}$$

to which there are three solutions corresponding to the $\pm$ states. Whilst (6.15) can be solved for numerically, a solution can be formed following the analysis given by [79], commencing by rewriting (6.15) in partial fraction form:

$$X + \frac{A}{(X-\rho)^2} + \frac{B}{(X+1-\rho)^2} = 0 \tag{6.16}$$

Considering the first case, in which $A = 1-\rho$ and $B = \rho$, by substituting $r_2 = \varepsilon_1$, $r_1 = 1+\varepsilon_1$ and $X = \rho - 1 - \varepsilon_1$, we obtain:

$$\varepsilon_1 + 1 - \rho + \frac{\rho - 1}{(1+\varepsilon_1)^2} - \frac{\rho}{(\varepsilon_1)^2} \tag{6.17}$$

which, ultimately results in a 5^{th} Order equation:

$$\varepsilon_1^5 + (3-\rho)\,\varepsilon_1^4 + (3-2\rho)\varepsilon_1^3 - \rho\varepsilon_1^2 - 2\rho\varepsilon_1 - \rho = 0 \tag{6.18}$$

from which a solution can be found as:

$$\varepsilon_1 = r_2 = \left(\frac{\rho}{3}\right)^{\frac{1}{3}}\left[1 + \frac{1}{3}\left(\frac{\rho}{3}\right)^{\frac{1}{3}} - \frac{1}{9}(\frac{\rho}{3})^{\frac{2}{3}} + \ldots\right] \tag{6.19}$$

Following a similar analysis for the second case, where $A = 1-\rho$ and $B = -\rho$, and substituting $r_2 = \varepsilon_2$, $r_1 = 1-\varepsilon_2$ and $X = \rho - 1 + \varepsilon_2$, we obtain:

$$\varepsilon_2 = r_2 = \left(\frac{\rho}{3}\right)^{\frac{1}{3}} \left[1 - \frac{1}{3}\left(\frac{\rho}{3}\right)^{\frac{1}{3}} - \frac{1}{9}\left(\frac{\rho}{3}\right)^{\frac{2}{3}} + \ldots\right] \tag{6.20}$$

For the third and final case, where $A = \rho - 1$ and $B = -\rho$, substitution of $r_1 = \varepsilon_3 = X - \rho$, $r_2 = 1 + \varepsilon_3$ and $X = \rho + \varepsilon_3$, we obtain the intermediary equation:

$$\varepsilon_3^5 + (2+\rho)\,\varepsilon_3^4 + (1+2\rho)\,\varepsilon_3^3 - (1-\rho)\,\varepsilon_3^2 - 2\,(1-\rho)\,\varepsilon_3 - (1-\rho) = 0 \tag{6.21}$$

which, due to its proximity to unity, is best re-written using $\eta = \varepsilon_3 - 1$, yielding:

$$\eta^5 + (7+\rho)\,\eta^4 + (19+6\rho)\,\eta^3 - (24+13\rho)\,\eta^2 + 2\,(6+7\rho)\,\eta + 7\rho = 0 \tag{6.22}$$

A series solution to Equation 6.22 in powers of $\nu = \frac{7\rho}{12}$ is:

$$\eta = -\nu\left(1 + \frac{23}{84}\nu^2 + \frac{23}{84}\nu^3 + \frac{761}{2352}\nu^4 + \frac{3163}{7056}\nu^5 + \frac{30703}{49392}\nu^6\right) + O(\nu^8) \tag{6.23}$$

From which $\varepsilon_3 = r_1$ can be found by application of the previous condition. It is now possible to specify the non-dimensional (X,Y,Z) position of the libration points for the general CRTBP, which vary according to the system mass ratio ρ. A table of the non-dimensional libration points for the Sun-Earth/Moon system, where $\rho = 3.0401 \times 10^{-6}$ and the coordinate system shown in Figure 6.1, calculated using the above equations is given in Table 6.2.1.

Libration Point	Sun-Earth System		
	X	Y	Z
L_1	-0.99	0	0
L_2	-1.01	0	0
L_3	1	0	0
L_4	-0.5	$\frac{\sqrt{3}}{2}$	0
L_5	-0.5	$-\frac{\sqrt{3}}{2}$	0

Tab. 6.1: Table non-dimensional libration points for the Sun-Earth/Moon system, $\rho = 3.0401 \times 10^{-6}$

6.2.2 Stability Analysis of Libration Points

A linear stability analysis of the libration points can be made with the simple derivation found in numerous texts [99, 79, 118]. The most concise derivation can be found within [118], which uses a change of variables $(\varepsilon, \eta, \zeta)$ such that:

$$\begin{aligned} \varepsilon &= X - X_{L_i} \\ \eta &= Y - Y_{L_i} \\ \zeta &= Z - Z_{L_i} \end{aligned}$$

within which the subscript L_i denotes a reference to libration point i. The resulting linear variational equations for motion about L_i are then given as:

$$\begin{aligned} \ddot{\varepsilon} - 2\dot{\eta} &= U^*_{xx}\varepsilon + U^*_{xy}\eta + U^*_{xz}\zeta \\ \ddot{\eta} + 2\dot{\varepsilon} &= U^*_{yx}\varepsilon + U^*_{yy}\eta + U^*_{yz}\zeta \\ \ddot{\zeta} &= U^*_{zx}\varepsilon + U^*_{zy}\eta + U^*_{zz}\zeta \end{aligned}$$

where $U^*_{ij} = U_{ij}|_{L_i}$. Forming a six-dimensional state vector, $\bar{\varepsilon} \equiv \left[\varepsilon, \eta, \zeta, \dot{\varepsilon}, \dot{\eta}, \dot{\zeta}\right]^T$, the variational equations can be written in state space form as:

$$\dot{\bar{\varepsilon}} = \mathbf{A}\bar{\varepsilon} \tag{6.24}$$

where $\mathbf{A}$ has the same form as that for (6.11) with the U_{ij} evaluated with respect to the libration point being considered.

Finding the eigenvalues of (6.24) enables determination of the stability for the linear system about a libration point; this infers information of the nonlinear system and the qualitative nature of the motion.

The eigenvalues corresponding to the equilateral libration points with $\rho \leq 0.0385$ or $\rho \geq 0.961$ are stable. Such stability in these orbital locations can be observed in the natural world through the Trojan asteroids[3], which are located in the equilateral libration points of the Sun-Jupiter system; the system ratio for the Sun-Earth/Moon system is 0.01215 and so the equilateral libration points for this system are stable.

Assessing (6.24) for the Sun-Earth/Moon system at libration points yields six eigenvalues: two real (one of which negative) and four purely imaginary. It is therefore mathematically possible to select initial conditions which only excite the oscillatory modes

[3] Following the determination of the quasi-stable triangular libration point locations, Lagrange predicted the existence of the Trojan asteroids of the Sun-Jupiter system. These were first observed by E.E. Barnard in 1904, some 134 years after the prediction was made.

and hence generate a stable periodic motion: this is the fundamental statement for the existence of libration point orbits.

Such libration point orbits are of interest to the space community since they offer advantageous mission possibilities within a gravitationally beneficial environment: within [84] it is proposed that the L_2 point of the Earth-Moon system be used as a staging point for a relay satellite to enable continuous communications with the far side of the Moon[4] and the interior L_1 point of the Sun-Earth/Moon system is currently the operational orbit of SOHO mission which observes solar activity [119].

6.3 Libration Point Orbits

This Section will present the mathematical formulation of libration point orbits which has led to the realization of both Lissajous and halo orbits. Lissajous and halo orbits are both a class of libration point orbit, the latter being distinguished by its periodic nature with a continually repeating orbit track within the synodic frame; Lissajous orbits are quasi-periodic, resulting from the difference in the in-plane and out-of-plane frequencies.

To achieve a truly periodic orbit, it is generally required to perform an orbital correction in order to achieve equal in-plane and out-of-plane frequencies; within the literature this is referred to as period or frequency control. Whilst this thesis will not consider the problem of frequency control for attaining halo orbits, it will be necessary to determine a halo orbit to enable the generation of a trajectory for agents to follow. This can be done via two methods: a Richardson approximation or a differentially corrected orbit. Both methods of obtaining a halo orbit will now be presented.

6.3.1 Richardson Approximation

The CRTBP cannot be expressed as a function of time, however approximations can be formed: in 1980 a third order analytical solution for periodic orbits about the collinear points was presented [116]. Whilst these results have been widely referenced and used within the community, including design of candidate ISEE3 orbits, they are actually inaccurate in the third-order term of the amplitude of the halo orbit [120].

The orthogonal coordinate system for the development of the Richardson approximation is located at one of the collinear libration points, with the x-y plane coinciding with the plane of motion of the primaries: the y-axis is direction in the motion direction of the

[4] This proposal was made during the time of the Apollo missions, subsequent to this time a world-wide agreement has been made to retain radio isolation of the far side of the moon.

rotating libration point and the x-axis is directed away from the larger primary. The z-axis completes the right-handed system. This coordinate system is shown in Figure 6.3.

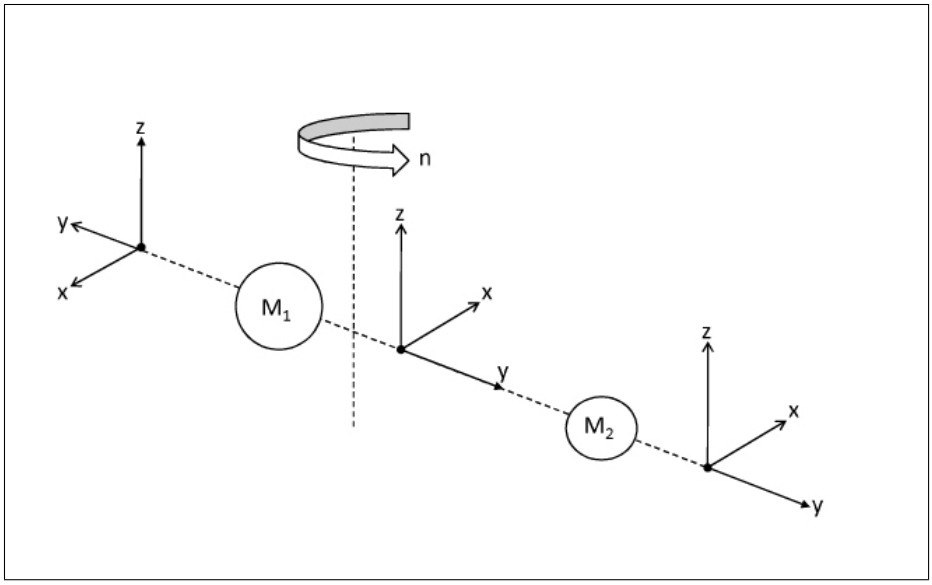


Fig. 6.3: The three possible libration point centered coordinate frames used for the development of Richardson's third order analytical approximation.

Determination of the approximate model stems from the formulation of the equations of motion for a particle moving in the vicinity of any of the collinear libration points. Upon the formulation of a normalised Lagrangian, a third order successive approximation is applied using the perturbation techniques of Lindstedt-Poincaré. A detailed derivation of the analytic solution can be found within [120]; the results to which are explicitly presented as:

$$\begin{aligned} x(t) &= a_{21}A_x^2 + a_{22}A_z^2 - A_x cos(\lambda t) + (a_{23}A_x^2 - a_{24}A_z^2)cos(2\lambda t) \\ &\quad + (a_{31}A_x^2 - a_{32}A_x A_z^2)cos(3\lambda t) \end{aligned} \tag{6.25a}$$

$$\begin{aligned} y(t) &= KA_x sin(\lambda t) + (b_{33}A_x^3 + b_{34}A_x A_z^2 - b_{35}A_x A_z^2)sin(\lambda t) \\ &\quad + (b_{21}A_x^2 - b_{22}A_z^2)sin(2\lambda t) + (b_{31}A_x^3 - b_{32}A_x A_z)sin(3\lambda t) \end{aligned} \tag{6.25b}$$

$$z(t) = A_z cos(\lambda t) + d_{21}A_x A_z cos(2\lambda t - 3) + (d_{32}A_z A_x^2 - d_{31}A_z^3)cos(3\lambda t) \tag{6.25c}$$

for which all the coefficients are given within Appendix B.

Upon implementation, (6.25) yields a third order approximation to the path of a periodic halo orbit about one of the libration points. An example orbit generated through this

technique, based on the implemented ISEE3 orbit and generated through the parameters encapsulated within Table 6.3.1, is shown within Figure 6.4.

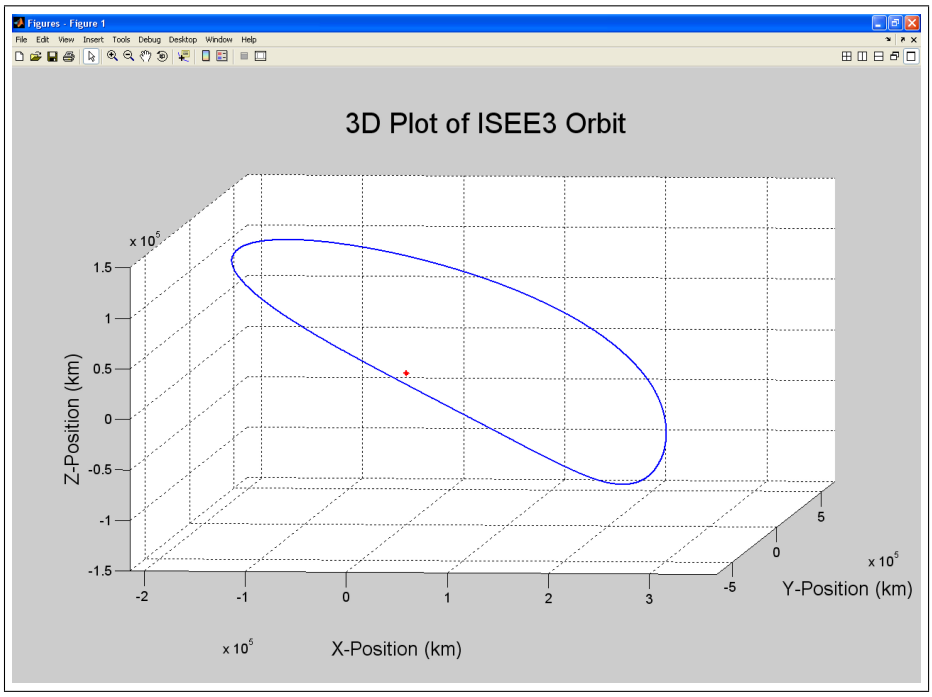


Fig. 6.4: ISEE3 orbital path (blue line) about the L_1 point (red point), generated through an analytical Richardson approximation, based on the Richardson constants given within Table 6.3.1

Whilst the Richardson approximation to a halo orbit is of third order, it is still not a precise prescription of a true halo orbit. If the initial conditions generated by the analytic method (corresponding to a crossing of the x,z-plane at $y = 0$ and $t = 0$ with zero z velocity) were to be placed into a dynamical model representative of the non-linearized CRTBP, the orbital path would quickly degrade from that of a halo orbit. Such an instance is shown in Figure 6.5 where the true orbital path is seen to diverge from a halo orbital track. The divergence is initially observed at the second crossing of the $y = 0$ axis, where in theory the crossing should be perpendicular.

6.3.2 Differentially Corrected Orbits

A differentially corrected halo orbit is a numerical targeting method, using Newton's method and the Richardson analytical approximation as an initial seed to allow for con-

A_x	2.06×10^5km	b_{32}	0.023019827
A_z	1.1×10^5km	b_{33}	-2.8451
K	3.22927	b_{34}	-2.3021
λ	2.086453455	b_{35}	-1.8704
a_{21}	2.092695581	d_{21}	-0.346865461
a_{22}	0.248297670	d_{31}	0.019043870
a_{23}	-0.905964795	d_{32}	0.398095425
a_{24}	-0.104464116	c_2	4.06107
a_{31}	0.793820195	c_3	3.0201
a_{32}	0.008268539	c_4	3.03054
b_{21}	-0.492445875	ω	0.98505017
b_{22}	0.060746467	Δ	0.2922144542
b_{31}	0.885700776		.

Tab. 6.2: Table of Richardson constants used within (6.25) for the analytic approximation to the ISEE3 mission orbit and resulting in Figure 6.4

vergence. This method was first applied within [117].

The method involves taking the Richardson analytical approximation at $t = t_0$ as an estimate of the initial state for a halo orbit: this location is converted from the Richardson frame, which has an origin based at the libration point of interest, to that of the global frame used in the representation of (6.11), which has the origin based at the system barycenter and is denoted by use of upper-case letters. The distinction between coordinate systems can be seen by comparing the X-axis values to Figures 6.4 and 6.5.

The Richardson estimate is located on the X,Z-plane with a component of velocity only in the positive y direction. Assuming that the orbit crosses the X,Z-plane at two separate points, the second point will be half way through the orbit at a time denoted by $t = t_{\frac{1}{2}}$ with corresponding state parameters at this time denoted by the subscript $\frac{1}{2}$. At this point we are seeking an initial state which will result in a perpendicular crossing, only having a component of velocity in the negative y direction. If under the initial conditions we result in a perpendicular crossing of the X,Z-plane at $t = t_{\frac{1}{2}}$ then the initial conditions are part of the periodic orbit; if not then the initial conditions must be modified in some way in order to drive $\dot{X}$ and $\dot{Z}$ to zero at the crossing. Performing such modifications is equivalent to solving a system of two equations $\left(\dot{X}_{\frac{1}{2}} \;\&\; \dot{Z}_{\frac{1}{2}}\right)$ in three unknowns (X_0, Z_0 & $\dot{Y}_0$). Although a minimum norm solution can be applied, it is more desirable to fix one of the initial state elements and so yield more control over the periodic orbit which

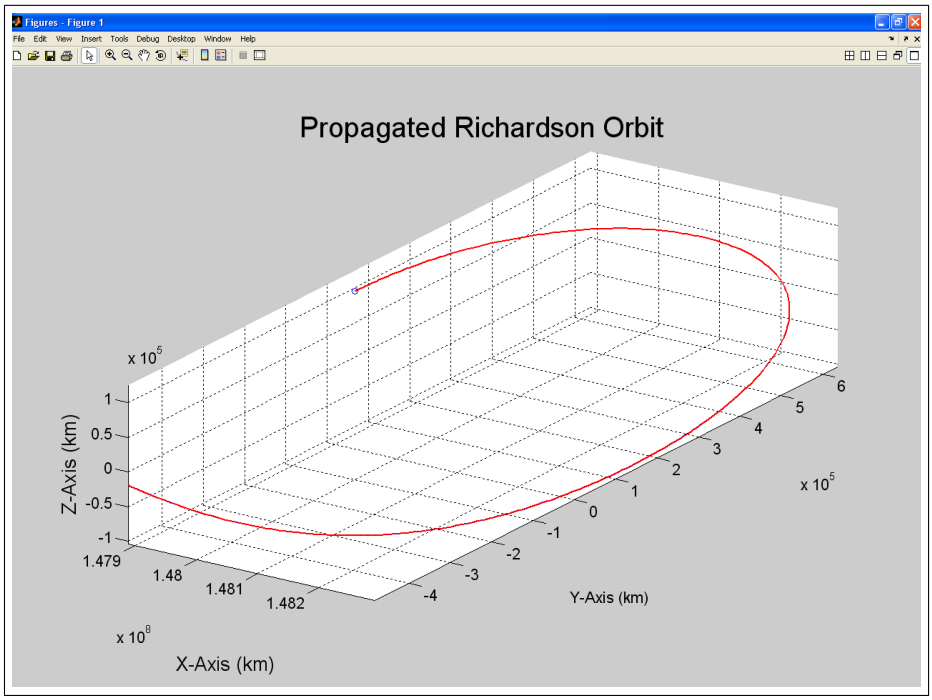


Fig. 6.5: ISEE3 initial orbital state (blue circle) propagated within representative dynamical model: the true orbital path (red line) shows rapid divergence from the analytically predicted path shown in Figure 6.4.

is generated by solving a system of two equations in two unknowns. This involves an iterative targeting method within which modification to the initial conditions is performed using the state transition matrix (STM[5]). The STM is a matrix, $\boldsymbol{\phi}(t_f, t_0)$, whose product with the state vector at an initial time t_0 gives the state vector at some future time t_f, which we can denote as $x(t_f) = \boldsymbol{\phi}(t_f, t_0)x(t_0)$.

For the dynamical system $\dot{\mathbf{x}}(t) = f(\mathbf{x})$ evaluated from t_0 to some future time t_f, the final state differential at t_f is given by

$$d\mathbf{x}_f = \boldsymbol{\phi}(t_f, t_0)\delta\mathbf{x}_0 + \dot{\mathbf{x}}_f dt_f$$

where the state transition matrix satisfies

$$\begin{aligned} \dot{\boldsymbol{\phi}}(t, t_0) &= \mathbf{F}(\mathbf{x}, t)\boldsymbol{\phi}(t, t_0), \\ \boldsymbol{\phi}(t_0, t_0) &= \mathbf{I}_{(6,6)} \end{aligned}$$

[5] The state transition matrix is also referred to as a fundamental solution matrix in some literature.

and **F** is the Jacobian of the vector field used as the state propagation matrix,

$$\mathbf{F}(\mathbf{x},t) \;=\; \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}$$

Using the method presented within [121] and [122], a differential correction process for a non-linear system of the form $\mathbf{e} = \mathbf{e}(\mathbf{z})$ can be used to find the parameter vector $\mathbf{z}$ which achieves a desired value of $\mathbf{e}$. The correction

$$\delta\mathbf{z} \;=\; \mathbf{A}^{-1}\delta\mathbf{e} \tag{6.26}$$

can be used iteratively until the norm of $\mathbf{e}$ is within a specified tolerance. Within this notation $\mathbf{A}$ is the Jacobian matrix of the system and applied to our system where we wish to retain a maximum A_x amplitude and alter values of $\dot{Y}_0$, Z_0 and $t_{\frac{1}{2}}$. The perturbation vectors and the Jacobian matrix for the problem are:

$$\begin{aligned}
\delta\mathbf{z} &= \begin{pmatrix} \delta Z_0 \\ \delta \dot{Y}_0 \\ dt_{\frac{1}{2}} \end{pmatrix} \\
\delta\mathbf{e} &= \begin{pmatrix} \delta Y_{\frac{1}{2}} \\ \delta \dot{X}_{\frac{1}{2}} \\ \delta \dot{Z}_{\frac{1}{2}} \end{pmatrix} \\
\mathbf{A} &= \begin{bmatrix} \phi_{(2,3)} & \phi_{(2,5)} & \dot{Y}_{\frac{1}{2}} \\ \phi_{(4,3)} & \phi_{(4,5)} & \ddot{X}_{\frac{1}{2}} \\ \phi_{(6,3)} & \phi_{(6,5)} & \ddot{Z}_{\frac{1}{2}} \end{bmatrix}
\end{aligned}$$

within which $\phi_{(i,j)}$ is the i, j element of $\boldsymbol{\phi}$ given explicitly for the CRTBP as:

$$\boldsymbol{\phi}(t_{\frac{1}{2}},t_0) = \begin{bmatrix}
\left(\frac{\partial X_{\frac{1}{2}}}{\partial X_0}\right) & \left(\frac{\partial X_{\frac{1}{2}}}{\partial Y_0}\right) & \left(\frac{\partial X_{\frac{1}{2}}}{\partial Z_0}\right) & \left(\frac{\partial X_{\frac{1}{2}}}{\partial \dot{X}_0}\right) & \left(\frac{\partial X_{\frac{1}{2}}}{\partial \dot{Y}_0}\right) & \left(\frac{\partial X_{\frac{1}{2}}}{\partial \dot{Z}_0}\right) \\
\left(\frac{\partial Y_{\frac{1}{2}}}{\partial X_0}\right) & \left(\frac{\partial Y_{\frac{1}{2}}}{\partial Y_0}\right) & \left(\frac{\partial Y_{\frac{1}{2}}}{\partial Z_0}\right) & \left(\frac{\partial Y_{\frac{1}{2}}}{\partial \dot{X}_0}\right) & \left(\frac{\partial Y_{\frac{1}{2}}}{\partial \dot{Y}_0}\right) & \left(\frac{\partial Y_{\frac{1}{2}}}{\partial \dot{Z}_0}\right) \\
\left(\frac{\partial Z_{\frac{1}{2}}}{\partial X_0}\right) & \left(\frac{\partial Z_{\frac{1}{2}}}{\partial Y_0}\right) & \left(\frac{\partial Z_{\frac{1}{2}}}{\partial Z_0}\right) & \left(\frac{\partial Z_{\frac{1}{2}}}{\partial \dot{X}_0}\right) & \left(\frac{\partial Z_{\frac{1}{2}}}{\partial \dot{Y}_0}\right) & \left(\frac{\partial Z_{\frac{1}{2}}}{\partial \dot{Z}_0}\right) \\
\left(\frac{\partial \dot{X}_{\frac{1}{2}}}{\partial X_0}\right) & \left(\frac{\partial \dot{X}_{\frac{1}{2}}}{\partial Y_0}\right) & \left(\frac{\partial \dot{X}_{\frac{1}{2}}}{\partial Z_0}\right) & \left(\frac{\partial \dot{X}_{\frac{1}{2}}}{\partial \dot{X}_0}\right) & \left(\frac{\partial \dot{X}_{\frac{1}{2}}}{\partial \dot{Y}_0}\right) & \left(\frac{\partial \dot{X}_{\frac{1}{2}}}{\partial \dot{Z}_0}\right) \\
\left(\frac{\partial \dot{Y}_{\frac{1}{2}}}{\partial X_0}\right) & \left(\frac{\partial \dot{Y}_{\frac{1}{2}}}{\partial Y_0}\right) & \left(\frac{\partial \dot{Y}_{\frac{1}{2}}}{\partial Z_0}\right) & \left(\frac{\partial \dot{Y}_{\frac{1}{2}}}{\partial \dot{X}_0}\right) & \left(\frac{\partial \dot{Y}_{\frac{1}{2}}}{\partial \dot{Y}_0}\right) & \left(\frac{\partial \dot{Y}_{\frac{1}{2}}}{\partial \dot{Z}_0}\right) \\
\left(\frac{\partial \dot{Z}_{\frac{1}{2}}}{\partial X_0}\right) & \left(\frac{\partial \dot{Z}_{\frac{1}{2}}}{\partial Y_0}\right) & \left(\frac{\partial \dot{Z}_{\frac{1}{2}}}{\partial Z_0}\right) & \left(\frac{\partial \dot{Z}_{\frac{1}{2}}}{\partial \dot{X}_0}\right) & \left(\frac{\partial \dot{Z}_{\frac{1}{2}}}{\partial \dot{Y}_0}\right) & \left(\frac{\partial \dot{Z}_{\frac{1}{2}}}{\partial \dot{Z}_0}\right)
\end{bmatrix} \tag{6.27}$$

It is worth noting that numerical computation of the STM is required since there is no analytic expression for $\boldsymbol{\phi}(t,t_0)$.

Using the correction method presented within (6.26) and the initial conditions for the *ISEE*3 mission given by the Richardson approximation, a differentially corrected orbit can be generated. Figure 6.6 shows the iterative modification of the seed state to produce a perpendicular x,z-plane crossing under the constraint of maintaining the A_x magnitude. Figure 6.7 shows a complete orbit resultant from application of the modified initial state.

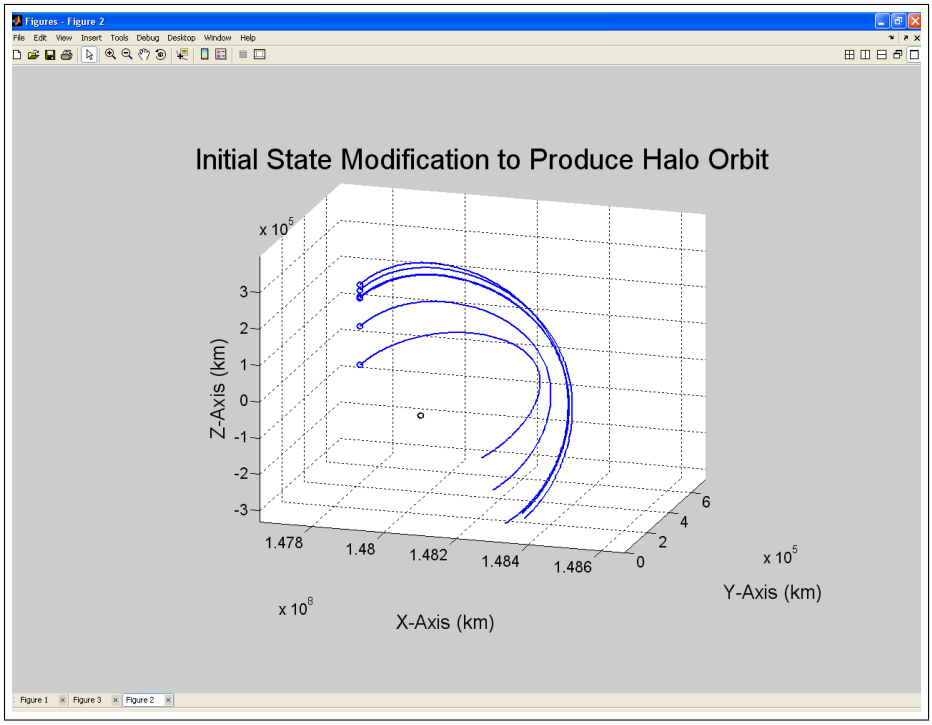


Fig. 6.6: Sequential modification of the ISEE3 seed state, produced by a Richardson third order approximation, to obtain a perpendicular crossing of the x,z-plane, whilst fixing the A_x amplitude at 206,000 km.

Once a differentially corrected halo orbit has been produced, the initial conditions to generate a halo orbit are known. If the STM for this periodic orbit is evaluated for a time equal to an orbital period, the STM produced is known as the monodromy matrix: the eigenvalues of this monodromy matrix (known as Floquet multipliers) dictate the stability of the periodic orbit. Whilst this still does not allow us to implement any results in an analytical fashion, using a continuous time orbital propagation, states along the orbital path can be saved at nominated discrete time instances and stored for 'look-up' at a later stage.

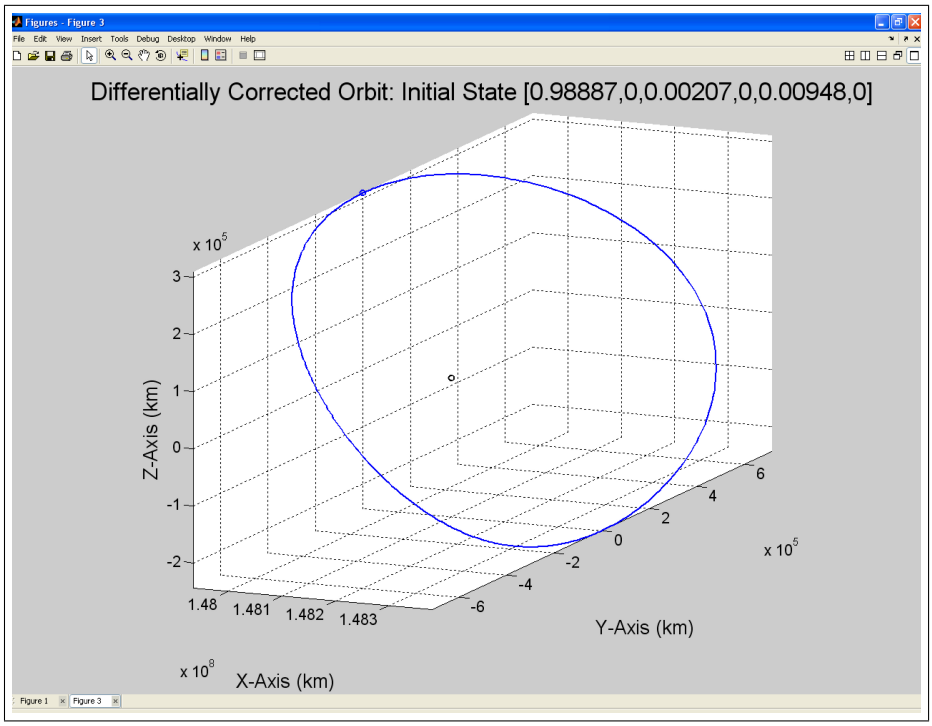


Fig. 6.7: Complete propagation of the differentially corrected orbit for the ISEE3 mission (blue line) resultant from the modified initial state (blue circle). The libration point is denoted by a black circle.

This will produce reference states of a halo orbit for use in discrete time simulations.

6.4 Chapter Summary

This Chapter has presented the dynamical regime and equilibrium solutions for the circular restricted three body problem. The existence of bounded trajectories about the equilibrium solutions were proven and methods for developing closed trajectories to result in halo orbits were presented using both analytical and numerical methods.

Using the material within this Chapter, not only is it possible to generate a desired trajectory for our agent system to follow, but it is possible to create a simulation environment wherein an agent is subject to the dynamical influence of the CRTBP dynamics. The material presented within this Chapter will also be used as a means to generate the metric on which to base decisions: this is to be detailed within Chapter 11. It should be noted that the development presented within this Chapter has only considered the circular

restricted three body problem: not that of the elliptic restricted three body problem, nor the inclusion of solar radiation pressure; these effects were not deemed pertinent to the overall aims of this thesis.

It remains to present the consequences of agent control action to the motion of the physical agent body in six degrees of freedom and this will be the topic of Chapter 7.

7. SPACECRAFT AGENT DYNAMICS

This Chapter presents the required knowledge for modeling agent motion in 6DoF, including definitions of coordinate reference frames and coordinate frame transformations. Equally, this Chapter can be viewed as how an agent can perceive and model the world in which it resides. Whilst the content of this Chapter is available within numerous classical texts such as [99, 123, 124], it is contained here for completeness.

The mathematical model of a rigid body in motion is described by dynamic and kinematic equations of motion: in our consideration, this body is a spacecraft agent. Dynamics relate forces and torques acting on the agent to the agents translational and angular velocity given within a specified frame of reference; kinematics provide integration of translational and angular velocities. For subsequent analysis, the agent will be assumed to be a rigid body which is holonomic with respect to control. Rotations will be based on the quaternion attitude representation, which allows for singularity free rotations [125].

Upon introduction of the pertinent coordinate frames for this investigation, the rigid body kinematics and dynamics for both rotation and translation (accounting for coordinate frame rotation) will be presented. These will then be combined into a six degree of freedom state space equation relating body frame percepts to 'real world' motions.

7.1 Spacecraft Agent Coordinate Frames

For the purposes of agent kinematics, it is assumed here that a spacecraft agent will exist in a single regime within which only the agents percepts are of direct use. The subsequent derivations related to agent dynamics will utilize three cartesian coordinate systems: the agent body frame, the lead agent frame and the agent world frame. Each of these frames are given by a set of three orthonormal basis vectors that obey the right hand rule, as shown within Figure 7.1. An additional frame of reference, which although not used within future derivations is applicable during mission operation, is the inertial geocentric frame of reference.

Agent Body Frame Denoted by $\mathcal{F}_{nb}$, the agent body frame of reference is a three-axis right handed orthogonal coordinate system built upon the principal axes of inertia

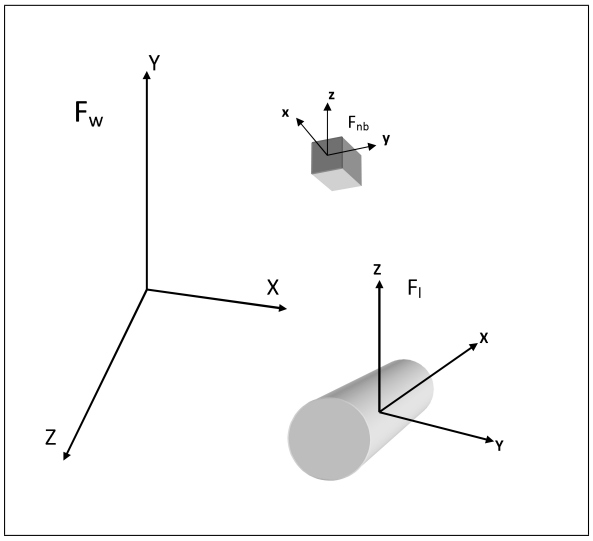


Fig. 7.1: Figure depicting the three frames of reference used: agent body (F_{nb}), lead agent (F_l) and world (F_w).

to agent n, with the origin placed at the center of mass of the agent.

Lead Agent Frame Denoted by F_l, the lead agent frame of reference is a body frame, with the origin placed at the center of mass of the lead agent.

Agent World Frame Denoted by F_w, the agent world frame of reference is the three-axis right handed orthogonal coordinate system used to represent the CRTBP. The z axis is directed perpendicular to the rotation plane of the Sun-Earth system, with the x axis directed from the system barycenter to the Earth; the y axis completes a right handed orthogonal frame.

Inertial Heliocentric Frame Denoted by F_h, the heliocentric frame of reference (also known as the Copernicus frame) is an inertial frame of reference originating from the center of mass of the Sun. The ecliptic plane (in this instance) is taken as a principal plane and a third axis is taken as being normal to this plane. In some cases the principal plane is taken as that parallel to the Earth's equatorial plane, though this is not applied here. One of the axes contained within the principal plane is in the direction of the vernal equinox.

Whilst other coordinate frames exist, such as the Earth based geocentric frame, these are not pertinent within this investigation and shall be omitted from future discussion.

7.2 Body Frame Transformations

The agent body frame, F_{nb}, is related to F_l and to F_w by a translation and rotation. For three axis orthogonal coordinate systems, translation in 3-space is uniformly represented as a direction vector $[x,y,z]^T$. Coordinate frame orientations and relative orientations, although intuitively visible as a sequence of rotations, are most efficiently represented using the 4-parameter quaternion notation often referred to as the Euler parameters. The quaternion is a computationally efficient and singularity free method of dealing with rotations; consequently it is now the standard method for dealing with attitude transformations [125]. A unit quaternion corresponding to a rotation between frames i and j is given by:

$$\mathbf{q}_{ij} = \begin{bmatrix} \kappa \\ \iota \end{bmatrix} \tag{7.1}$$

where the vector $\kappa \in \Re^3$ and scalar $\iota \in \Re$ components are constrained by $\iota^2 + \kappa^T \kappa = 1$.

From Euler's theorem of rotational displacement, any rotation of an angle α about an axis denoted by the unit vector $\mathbf{e} \in \Re^3$, known as the Euler axis or eigenvector of rotation, may be described by the unit quaternion $\mathbf{q}$ as:

$$\kappa = \mathbf{e} sin\left(\frac{\alpha}{2}\right) \tag{7.2}$$

$$\iota = cos\left(\frac{\alpha}{2}\right) \tag{7.3}$$

The Rodriguez formula presents the direction cosine matrix describing rotations between frames i and j expressed in terms of the quaternion parameters as

$$[\mathbf{A}(\mathbf{q_{ij}})] = \mathbf{R}_{ij} = (\iota^2 - \kappa^T \kappa) I_{3x3} + 2\kappa^T \kappa - 2\iota \lfloor \kappa^{\times} \rfloor \tag{7.4}$$

where $\lfloor \kappa^{\times} \rfloor$ represents the antisymmetric cross product matrix, operating on vector κ and is given by

$$\lfloor \kappa^{\times} \rfloor = \begin{bmatrix} 0 & -\kappa_z & \kappa_y \\ \kappa_z & 0 & -\kappa_x \\ -\kappa_y & \kappa_x & 0 \end{bmatrix} \tag{7.5}$$

when $\kappa = [\kappa_x \ \kappa_y \ \kappa_z]^T$.

In the context of agent perceptions, to obtain an attitude transformation, a quaternion multiplication is performed. For subsequent analysis the symbol $\otimes$ will be used to denote the quaternion multiplication and for two unit quaternions, $\mathbf{q}^1$ and $\mathbf{q}^2$, it is defined in vector form as:

$$\mathbf{q}^1 \otimes \mathbf{q}^2 = \begin{bmatrix} q_4^2 & q_3^2 & -q_2^2 & q_1^2 \\ -q_3^2 & q_4^2 & q_1^2 & q_2^2 \\ q_2^2 & -q_1^2 & q_4^2 & q_3^2 \\ -q_1^2 & -q_2^2 & -q_3^2 & q_4^2 \end{bmatrix} \begin{bmatrix} q_1^1 \\ q_2^1 \\ q_3^1 \\ q_4^1 \end{bmatrix} \tag{7.6}$$

The quaternion multiplication given within (7.6) can be considered in two ways. Firstly if the quaternion components of two successive rotations are known, the composite rotation $\mathbf{R}_{ac} = \mathbf{R}_{ab}\mathbf{R}_{bc}$ can be computed by $\mathbf{q}_{ac} = \mathbf{q}_{ab} \otimes \mathbf{q}_{bc}$. Alternatively, the difference between a current quaternion vector $\mathbf{q}_1$ and another quaternion vector $\mathbf{q}_2$ can be computed by $\mathbf{q}_1^{-1} \otimes \mathbf{q}_2$. Where $\mathbf{q}^{-1} = [-\kappa\ \iota]^\top$ represents the inverse quaternion corresponding to the reverse rotation of α about $\mathbf{e}$ and $[0^\top\ 1]^\top \in \Re^4$ represents the identity relation. It should be pointed out here that quaternion multiplication is not commutative, that is $\mathbf{q}_{ab} \otimes \mathbf{q}_{bc} \neq \mathbf{q}_{bc} \otimes \mathbf{q}_{ab}$.

7.3 Equations of Motion

In the following Section, the kinematic differential equations and dynamic equations of motion for both rotation (based upon the quaternion notation) and position vectors will be presented.

Throughout this Section and for the remainder of this thesis, the following notation for all measurements will be retained: superscripts indicate which frame of reference the parameter corresponds to; subscripts indicate the two frames of reference to which a measurement parameter correspond. For instance, if θ represents a generic parameter, then θ_i^j indicates parameter measurement within frame i with respect to frame j.

7.3.1 Rotational Kinematics and Dynamics

Here we are considering the rotational motion of an agent in free space relative to a lead agent. Differential rotational kinematics describe how the attitude of a fixed coordinate system evolves on SO(3)[1] with respect to another coordinate system. This can be visualized as the ability to describe the rotational motion of the agent body frame within either the lead agent or world frames of reference. When parameterized in terms of the

[1] SO(3) here denotes a rotation group, which is defined as the group of all rotations about the origin of three dimensional Euclidean space $\Re^3$. The group of rotation matrices comprises of 3x3 orthogonal matrices with a determinant equal to 1; the resulting coordinate transformation or rotation will change the direction of the transformed vector without affecting its length [126].

quaternion vector, the differential rotational kinematics are given as:

$$\dot{\mathbf{q}}_i = \frac{1}{2}\Omega(\omega_i^j)\mathbf{q}_i \tag{7.7a}$$

$$= \frac{1}{2}\begin{bmatrix} \omega_i^j \\ 0 \end{bmatrix} \otimes \mathbf{q}_i \tag{7.7b}$$

$$= \frac{1}{2}\begin{bmatrix} -\lfloor \omega_i^j \times \rfloor & (\omega_i^j)^T \\ -\omega_i^j & 0 \end{bmatrix} \mathbf{q}_i \tag{7.7c}$$

where ω_i^j is the angular velocity of frame i relative to the inertial frame j.

In traditional applications of strap-down inertial reference systems for aerospace vehicles, the body rates (ω) are measured by rate gyros and the kinematic differential equation (7.7) is integrated numerically to determine the orientation of the vehicle in terms of quaternions in a dead-reckoning approach. For purposes of application by our agent, it should be clarified that the agent will be sensing the angular velocity of its body frame, F_{nb}, relative to an external frame within which the body frame is moving.

Rotational dynamics relates the differential of the angular velocity to the current angular velocity and torques acting upon our rigid agent. Euler's momentum equation provides a solution for this, most commonly written in the form:

$$\mathbf{J}\dot{\omega} + \lfloor \omega \times \rfloor \mathbf{J}\omega = \sum \tau \tag{7.8}$$

where $\mathbf{J}$ is the inertial matrix and $\sum \tau$ represents the vector sum of all torques acting on the rigid body. Applied to agent n, the dynamical rotational equation can be given as

$$\mathbf{J}_n \dot{\omega}_i^j + \lfloor \omega_i^j \times \rfloor \mathbf{J}_n \omega_i^j = \tau_{nc} + \tau_{nd} \tag{7.9}$$

where here the control and disturbances torques acting on agent n, $\tau_{nc} \in \Re^3$ and $\tau_{nd} \in \Re^3$ respectively, have been made evident.

7.3.2 Linear Kinematics and Dynamics

Here we are considering the motion of an agent in free space relative to a lead agent. The dynamical equation can be augmented to include external perturbations though this will not be considered at this time. Whilst linear momentum and Newtonian mechanics can provide trivial solutions for the kinematics and dynamics of a rigid body, the addition of rotational motion to encompass the full 6DoF relative mechanics modifies the familiar

double integrator model (7.10), through dynamic coupling.

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{7.10}$$

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} x \tag{7.11}$$

If we let $\dot{\mathbf{a}}^i$ denote the time derivative of an arbitrary vector $\mathbf{a}$ measured in frame i, then the time derivative of $\mathbf{a}$ measured in frame j is given by:

$$\dot{\mathbf{a}}^j = \dot{\mathbf{a}}^i + \omega_i^j \times \mathbf{a}^i \tag{7.12}$$

which simply states that the rate of change of the vector $\mathbf{a}$ as observed in the fixed frame j equals the rate of change of the vector $\mathbf{a}$ as observed in the rotating coordinate system i with angular velocity ω, plus the vector product $\omega \times \mathbf{a}$ [123].

Applying (7.12) to interagent distance vector $d_i^j \in \Re^3$, measured in frame i, we obtain the kinematic equation:

$$\dot{\mathbf{d}}_i^j = \dot{\mathbf{d}}_i^j + \lfloor \omega \times \rfloor \mathbf{d}_i^j \tag{7.13}$$

Applying (7.12) to Newton's Second law of motion, $\sum \mathbf{f} = \frac{d}{dt}(m\mathbf{v})$, we obtain the modified dynamic equation for interagent velocities $\mathbf{v}_i^j \in \Re^3$ whilst accounting for rotational motion of the follower agent:

$$\sum \mathbf{f}_i^j = \frac{d}{dt}(m\mathbf{v}_i^j) + m\lfloor \omega \times \rfloor \mathbf{v}_i^j \tag{7.14}$$

7.3.3 Velocity Transformations in 6DoF Motion

Taking equations (7.7), (7.9), (7.13) and (7.14) we can formulate a six degree of freedom state space representation for the motion of one coordinate system with respect to another. Based upon the 13-dimensional state vector $[\mathbf{d}, \mathbf{v}, \mathbf{q}, \boldsymbol{\omega}]^T$ and considering the motion of coordinate systems i and j, measured within frame i and expressed in frame j, the state

space representation is:

$$\frac{d}{dt}\begin{bmatrix}\mathbf{d}_i^j\\ \mathbf{v}_i^j\\ \mathbf{q}_i^j\\ \omega_i^j\end{bmatrix} = \begin{bmatrix}\mathbf{v}_i^j + \lfloor\omega_i^j\times\rfloor\mathbf{d}_i^j\\ \lfloor\omega_i^j\times\rfloor\mathbf{v}_i^j\\ \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_i^j)\mathbf{q}_i^j\\ -\mathbf{J}_n^{-1}\lfloor\omega_i^j\times\rfloor\mathbf{J}_n\boldsymbol{\omega}_i^j\end{bmatrix} + \begin{bmatrix}0 & 0\\ m_n^{-1}I & 0\\ 0 & 0\\ 0 & \mathbf{J}_n^{-1}\end{bmatrix}\begin{bmatrix}\sum\mathbf{f}_i^j\\ \sum\boldsymbol{\tau}_i^j\end{bmatrix} \tag{7.15a}$$

$$= \begin{bmatrix}\lfloor\omega_{ij}^i\times\rfloor & I_{(3,3)} & 0_{(3,4)} & 0_{(3,3)}\\ 0_{(3,3)} & \lfloor\omega_{ij}^i\times\rfloor & 0_{(3,4)} & 0_{(3,3)}\\ 0_{(3,3)} & 0_{(3,3)} & \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_{ij}^i) & 0_{(3,3)}\\ 0_{(3,3)} & 0_{(3,3)} & 0_{(3,4)} & -\mathbf{J}_n^{-1}\lfloor\omega_{ij}^i\times\rfloor\mathbf{J}_n\end{bmatrix}\begin{bmatrix}\mathbf{d}_{ij}^j\\ \mathbf{v}_{ij}^j\\ \mathbf{q}_{ij}^j\\ \omega_{ij}^j\end{bmatrix} \tag{7.15b}$$

$$+\begin{bmatrix}0 & 0\\ m_n^{-1}I & 0\\ 0 & 0\\ 0 & \mathbf{J}_n^{-1}\end{bmatrix}\begin{bmatrix}\sum\mathbf{f}_i^j\\ \sum\boldsymbol{\tau}_i^j\end{bmatrix}$$

$$= f(\mathbf{x},t) + \mathbf{B}\mathbf{u}(t) + \mathbf{C}\mathbf{u}_{ext} \tag{7.15c}$$

within which $\sum\mathbf{f}_i^j$ and $\sum\tau_i^j$ represent the relative forces and torques, including both control and disturbance components, acting on the two reference frames.

The state space representation given within (7.15c) can be conceptually thought of as how agent n, with percepts relating to its body frame $\mathcal{F}_{nb}$ and here denoted by the superscript i, can model the world in which it resides, relating to either the lead agent or world frame of reference, $\mathcal{F}_l$ or $\mathcal{F}_w$ respectively, here denoted by the superscript j.

7.3.4 6DoF Motion Within the CRTBP

Chapter 6 has presented the dynamics of a particle under influence of the CRTBP. Taking care over the use of units, these equations can be augmented with (7.15c) to enable computation of the full 6DoF controlled motion within the CRTBP. This computation can be completed using three methods:

1. Direct implementation of the global nonlinear CRTBP model given within Equations (6.1), (6.2) & (6.3).

2. Implementation of the equations of motion linearized about a libration point of interest (6.24).

3. Relative motion from a point on the halo orbit.

Whilst the use of item 1 is obvious, items 2 and 3 are of relevance when considering multiple agents which are proximate and under the influence of the CRTBP regime. If we consider a multi-agent system with inter-agent distances measurable in meters; for our system D is 1.496×10^{11}meters[2]. Consequently rounding errors will be present within any computation of Equations (6.1) ,(6.2) & (6.3) when considering the inter-agent distance. The use of (6.24) removes this source of computation error.

The method suggested within item 3, and presented within [127], involves consideration of relative motion between a moving point on a specified halo orbit. This specified point can be considered as a lead agent prescribing the halo orbit, with another agent in proximity. Assuming the lead agent has a state $\mathbf{x}_{lw}^{w}$ in $\mathcal{F}_w$ and the proximate agent has a state $\mathbf{x}_{pw}^{w}$ in $\mathcal{F}_w$, defining $\delta\mathbf{x} = \mathbf{x}_{pw}^{w} - \mathbf{x}_{lw}^{w}$, then for small $\delta\mathbf{x}$ the following approximation for the disturbance differential can be made:

$$\delta\dot{\mathbf{x}}_{pw}^{w} = \mathbf{A}(\mathbf{x}_{lw}^{w}, t)\delta\mathbf{x}_{pw}^{w} \tag{7.16}$$

where $\mathbf{A}(\mathbf{x}_{lw}^{w}, t)$ is that from (6.11) evaluated for $\mathbf{x}_{lw}^{w}$.

Use of (7.16) is equivalent to (6.24) when considering computational expense, though in some respects it could be assumed that since (7.16) is based upon relative distances the agent can directly compute the resulting differential and hence predict future unforced motions. However (7.16) also requires precise knowledge of the lead agent state in $\mathcal{F}_w$ to enable computation of $\mathbf{A}(\mathbf{x}_{lw}^{w}, t)$. Chapter 4 has already presented the issue that accurate absolute position measurement within the world frame is not possible: consequently the parameters encapsulated within the $\mathbf{A}(\mathbf{x}_{lw}^{w}, t)$ matrix of (7.16) would be imprecise and any differential motion predicted could only be accurate to within a certain error bound. Whilst positional determination within $\mathcal{F}_w$ is problematic, this is not true for attitude determination which can be achieved to very high accuracy using current technology [123].

7.4 Chapter Summary

This Chapter has presented the dynamics and kinematics of motion for different coordinate frames with six degrees of freedom. This is the equivalent of being able to describe the motion of a body frame with respect to an external frame of reference.

Methods of combining the six degree of freedom equations with the dynamics of the CRTBP presented within Chapter 6 have been presented and these will form the basis of computer simulations which will be completed in latter Chapters.

Coordinate frames for use within the development of the equations of motion have been presented: these coordinate systems will play an additional role within future system

[2] This value of D is based upon the Sun-Earth/Moon system

development, within different mission phases, and so shall be summarized within Table 7.1.

Coordinate System	**Application Situation**
Agent Body Frame, $\mathcal{F}_{nb}$	All agent measurements regrading local state information are measured within this frame. Chapter 8 uses information expressed in this frame for purposes of state estimation.
Lead Agent Frame, $\mathcal{F}_{l}$	Chapter 9 produces filtered state information with respect to this frame and is used within Chapter 10.
Agent World Frame, $\mathcal{F}_{w}$	Any computation requiring knowledge of global position will invoke use of this frame, such instances involve situational use of (7.16) within Chapter 11.
Heliocentric Frame, $\mathcal{F}_{h}$	This frame may be used to specify a particular orientation in the celestial sphere such that the interferometer optics may be trained onto a desired observation source.

Tab. 7.1: Table of coordinate frames used and their application situation.

8. SPACECRAFT AGENT GUIDANCE

This Chapter will present guidance methodologies which can be implemented within autonomous control solutions. A brief overview of current guidance methods will be followed by a more in depth look at the use of potential fields. This Chapter will not consider methods of obtaining sensor data; only the use of relevant information to form a guidance solution.

8.1 Autonomous Guidance

Guidance can be defined as a solution to obtain the desired system goal or state. Within the context of robotic systems, this goal relates directly to the 'real world' rather than a system state which may exist computationally. Consequently, robotic system goals generally refer to a particular location or trajectory in 3-space and so the definition of guidance as: *the process of guiding the path of an object to a given point, which in general may be moving*, is valid [128].

Guidance is an essential part of any autonomous robotic system since without guidance, a system is clearly unable to reach a particular state. Moreover, guidance is intimately interrelated with control, as the control action is derived directly from the guidance system. It is arguable that the greatest resource for guidance laws exists within the literature relating to missile guidance, though complete documentation of all possible guidance solutions is far beyond the scope of this thesis and the interested reader is referred to the following survey papers: [129, 130] & [131], with an interesting personal narrative given within [132].

Guidance methods can be broadly categorized as either point or path tracking[1].

Point tracking The control objective for a point tracking situation, where a target point is denoted by $\mathbf{x}_p(t) \triangleq [x_p(t), y_p(t), z_p(t)]^\top \in \Re^3$ and which may or may not be moving,

[1] Within some literature tracking is referred to as regulation, though for clarity only 'tracking' will be used here.

relating to a current physical location $\mathbf{x}(t) \triangleq [x(t), y(t), z(t)]^\top \in \Re^3$ is to achieve:

$$\lim_{x \to \infty} (\mathbf{x}(t) - \mathbf{x}_p(t)) = 0, \tag{8.1}$$

The control methodology outputs some form of action, based upon guidance generated from current percepts relating to a pertinent task. Point tracking methods could be viewed as reactive since the guidance is generally based upon instantaneous percepts which results in an instantaneous value of guidance.

Path tracking Within path tracking, a guidance law is generated composed of both speed and direction: these are based upon deviation from a pre-generated path. The desired path (or trajectory) is generated (or planned) through some form of optimisation process based ultimately on the current and desired state, though may also include relevant criteria such as avoidance or scheduling. Whilst a path may be initially computed off-line, the path may, in general, be modified in real time to account for new immediate priorities or a changing environment for which the original path is no longer viable.

For the application of interferometry, it is desirable to regulate the individual elements to specific array-points and as such we seek to achieve a point tracking solution. Guidance methods for point tracking are abundant throughout literature, since this represents the most fundamental guidance solution: that based exclusively upon state error. It is important to consider the environment which the interferometer agents will reside and the operations which they will be tasked with. Each agent must be regulated to a specific point in 3-space, though numerous agents will be operating at the same time. Consequently in addition to point tracking, each agent must provide for the avoidance of inter-agent collisions. Chapter 4 touched upon the use of potential functions within the research area of robotics for the provision of online guidance inclusive of avoidance constraints: it is this method which will be used in part to provide guidance for the interferometer agents.

8.2 Potential Fields

Potential fields are a method of specifying a desired response, given current percepts. Within the method percepts are used within (possibly several) pre-defined function(s) that generate a kinematic field: the desired response is in the negative gradient of this field. Potential fields may be classified as a point tracking guidance method, since the output guidance is a function of current position and percepts. Relevant potential functions to develop a desired kinematic field in 3-space, for attraction and repulsion, are formulated

within [102, 103, 133, 134]. These functions can equally be considered as providing guidance for motion to a desired location in 3-space, and avoidance of specific locations in 3-space; both of which may be time varying.

8.2.1 Attraction

Given relevant percepts relating to the world within which the agent resides, a potential function to result in attraction to a perceived point in 3-space is given as a negatively weighted inter-distance:

$$\Upsilon_{gather_i}(t) \quad = \quad -a_p \mathbf{d}_p(t) \tag{8.2}$$

where a_p is a scalar weighting applied to $\mathbf{d}_p(t) = \mathbf{x}(t) - \mathbf{x}_p(t)$, where $\mathbf{x}_p(t)$ relates to a desired location. It is evident that that at large inter-distances, $\mathbf{d}_p(t)$, the potential function output will be high and upon reaching the desired point in 3-space, $\mathbf{x}_p(t)$, the contribution will be zero.

8.2.2 Repulsion

Formulation of avoidance relationships is marginally more complex than those for attraction, since the converse kinematic field is desired: repulsion is required at proximity to a specified location but should become vanishingly small when $\mathbf{d}_p(t)$ is large. This is achieved by using the following function:

$$\Upsilon_{avoid_p}(t) \quad = \quad -\mathbf{d}_p(t) \left\{ b_p \exp\left(\frac{-\|\mathbf{d}_p(t)\|}{c_p} \right) \right\} \tag{8.3}$$

within which $\mathbf{d}_p(t) = \mathbf{x}(t) - \mathbf{x}_p(t)$ and $\mathbf{x}_p(t)$ relates to a location to be avoided and b_p and c_p are scalar parameters prescribing the sphere of influence of the function **??**.

8.2.3 Complex Fields

Complex potential fields can be created by summation of the elemental potential functions (8.2) and (8.3) to form (8.4).

$$\Upsilon_{complex_i}(t) \quad = \quad \Upsilon_{gather_i}(t) + \Upsilon_{avoid_p}(t) \tag{8.4}$$

This is the essence of the guidance systems described within [102, 103, 133, 134]. However, such implementations can give rise to problems of local minima: undesired equilibrium locations within the kinematic field. This problem has been looked at by the community and solutions sought by using bifurcating, Laplacian and super-quadric potential fields. The latter of these is beneficial in that it is possible to encode dimensionality within the field and thus produce a potential field to account for angular motion in addition to position.

8.3 Chapter Overview

This Chapter has provided a brief overview of current autonomous guidance solutions and focussed upon the use of potential fields, used commonly within robot guidance. It is this guidance method that shall be exploited in subsequent Chapters.

Part III

AGENT SYSTEM DEVELOPMENT

9. SPACECRAFT AGENT STATE ESTIMATION

This Chapter will present a methodology for obtaining accurate relative state estimation within a multi-agent system operating in the 3-space environment. Using only agent percepts, determination of the full agent state is required: 'perfect' measurement will not be assumed and the agents must obtain full state estimation in the presence of realistic sensor noise and errors.

9.1 Problem Formulation

We are faced with the need to formulate a coordinate frame for a multi-agent system based upon local measurements. Only local measurements will be permitted since the need for high precision localization is assumed: current state of the art technology is insufficient for global position estimation.

The coordinate system implemented will be fixed to one of the agents, which will be termed the 'lead agent'. In doing this we are forming F_l, as presented within Chapter 7, to which all other agents will be related by their body frame coordinate system F_{nb}. The choice of F_l origin will be assumed from a central agent within the cluster[1].

Each agent measures the distance and bearing (azimuth and elevation) to all other agents, within its own local inertial coordinate system F_{nb}. All interagent data is relayed to a central agent, which uses a data fusion process to estimate the agent states relative to itself; each agent is in turn informed of its state within the lead agent coordinate frame F_l. Each agent is assumed to have knowledge of its own attitude with respect to the celestial sphere through use of star sensors but does not have any global position knowledge. Due to the centralization of the data and resulting positioning data being with respect to a central agent, the representative dynamical model is that of the double integrator (7.10) and not that of the combined dynamics given within (7.15c).

Figure 9.1 shows the state estimation process occurring, where the cylindrical shape represents the lead agent, to which the coordinate frame F_l is attached.

[1] In the implemented interferometry system, this coordinate frame origin relates to the central mothercraft which assumes the role of combiner after deployment of the interferometer agents.

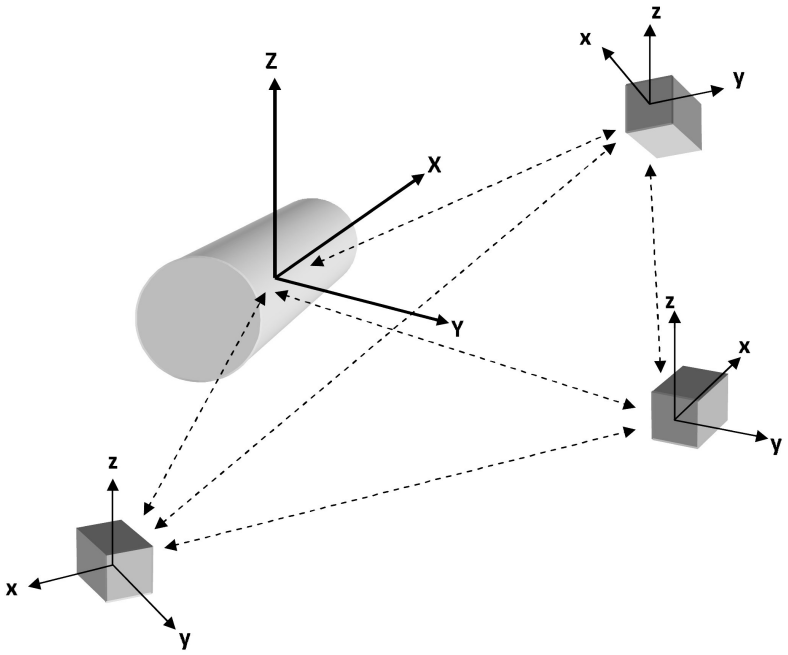


Fig. 9.1: System state through consensus: each agent measures the azimuth and elevation angles of each other agent in its local inertial coordinate system

Whilst implementation of a centralized state determination process could be considered a single point failure source, in considering a homogenous group of agents, it is not unreasonable to assume that any of the agents are capable of completing the data fusion process to obtain state estimation of the entire community. As such it is possible that at any given time a different agent within the community may be tasked with the data fusion process and maintain the agent society upon failure of its predecessor; thus removing the single point failure source. Indeed, the process involved in this state estimation could be considered as a software agent which is capable of being transferred or deployed upon any hardware agent.

9.2 Methodology

Irrespective of the implemented methodology in gathering inter-agent data, from a purely geometric point of view the following quantities can be relied upon as measured for an

agent group:

- Attitude of each agent in the inertial frame related to the stars.
- Distance between the center of mass of any two agents and its rate of change.
- Angular direction of any agent as viewed from any other agent and expressed as a direction vector in the local inertial coordinate system of the viewing agent.
- Translational and rotational accelerations of each agent relative to the inertial coordinate system.

In most practical systems only part of these quantities would be measured. The important points to note are:

- There are no other measured quantities to be relied upon.
- Even if all of the above quantities are measured simultaneously with some accuracy, the achievable accuracy of state measurement is limited to some portion of the individual accuracy as a result of data fusion.
- Positions cannot be estimated without bearing measurements and this implies that some kind of active or passive vision system is inevitable.
- Gravitational potential cannot be measured by any onboard system; such information can only be inferred from an on-board navigational computer using gravitational models and estimates of the current global position.

Here we assume the use of a vision system which is capable of providing both distance and bearing measurements. The signal processing required to achieve this is not entered upon since this is a routine implementation of existing techniques[2], what is of importance here is the way in which this information is utilized.

Assume that for N agents the positions of camera focus points are described in the agent group inertia coordinate system by $p_i = [x_i, y_i, z_i], i = 1,2,3,...N$. Then the ideal (not the measured) azimuth angles are defined by

$$\alpha_{ij} = \arcsin\left(\frac{z_j - z_i}{\|p_j - p_i\|}\right), i \neq j, i = 1,...,N; j = 1,...,N \tag{9.1}$$

[2] For relevant literature the interested reader is recommended [135] and [136]

and the elevation angles by

$$\beta_{ij} = \text{sign}(x_j - x_i) \arccos\left(\frac{y_j - y_i}{\|p_j - p_i\|}\right) \tag{9.2}$$

so that the range of each α_{ij} is $[-\pi/2, \pi/2]$ while the range of each β_{ij} is $[-\pi, \pi]$. Discontinuity of β at π and $-\pi$ is not a problem as the metric used for optimization of bearing angles is continuous.

For N agents, each camera will measure the azimuth $\widetilde{\alpha}_{ij} = \alpha_{ij} + \varepsilon_{ij}$ and elevation $\widetilde{\beta}_{ij} = \beta_{ij} + \eta_{ij}$, with some noise ε_{ij} and η_{ij} where $i \neq j$, $i = 1,2,3,\ldots,N$ and $j = 1,2,3,\ldots,N$. Assuming optimal calibration of the cameras being used, the quantization errors can be assumed independent and bounded; similar to rounding errors. For n agents, the total set of measurements $\{\widetilde{\alpha}_{ij}, \widetilde{\beta}_{ij}, \widetilde{d}_{ij}\}$ is $(N^2 - N)$, and this is assumed available.

Under the assumption of independently distributed measurement errors of *a priori* known variations $\sigma_\alpha^2, \sigma_\beta^2, \sigma_d^2$, the log likelihood function in terms of the agent coordinates is given by:

$$\begin{aligned} L(x_1, y_1, z_1, \ldots, x_N, y_N, z_N) &= \sum_{i \neq j} \log f_\alpha\left(\widetilde{\alpha}_{ij} - \alpha(p_i, p_j)\right) + \log f_\beta\left(\widetilde{\beta}_{ij} - \beta(p_i, p_j)\right) \\ &\quad + \log f_d\left(\widetilde{d}_{ij} - d_{ij}(p_i, p_j)\right) \end{aligned} \tag{9.3}$$

where $f_\alpha(w), f_\beta(w)$ and $f_d(w)$ are distribution functions of α, β and d measurement errors, respectively.

The $f_\beta(w)$ should be of the form $f_\beta(w) = f_\beta^-(\tilde{\rho}(w))$ with a periodic function $\tilde{\rho}(w) = \tilde{\rho}(w + 2\pi), \ \forall w$, so that the discontinuity of the azimuth at π to $-\pi$ does not cause any problems. Minimization of L for $x_1, y_1, z_1, ..., x_n, y_n, z_n$ provides the maximum likelihood estimates of satellite positions for given camera based measurements $\hat{d}_{ij}$, $\hat{\alpha}_{ij}$, $\hat{\beta}_{ij}$.

Figure 9.2 shows in block diagram form the operation of the vision system. Figure 9.3 illustrates the average and maximum errors for coordinate estimates of three satellite agents over 100 repetitions of non-linear least squares (NLS) estimation, performed by sequential quadratic programming, applied to distance and bearing information where $\alpha^e = \beta^e = 1.5^o$ and $d^e = 5mm$ were selected to be representative errors for the implemented vision system. Note that over 50m a 1.5^o bearing error leads to a 1.3m position error: increased accuracy in coordinate estimation can only be achieved through the use of higher resolution hardware[3] or the availability of a larger data set, which in this case would correspond to the use of more agents within the agent system.

[3] Using an imaging device with a 60° field of view and 640x480 resolution will result in a $50m$ distant $1m^2$ target being represented by a maximum of 92 pixels; this will reduce with increasing observation distance and/or angle.

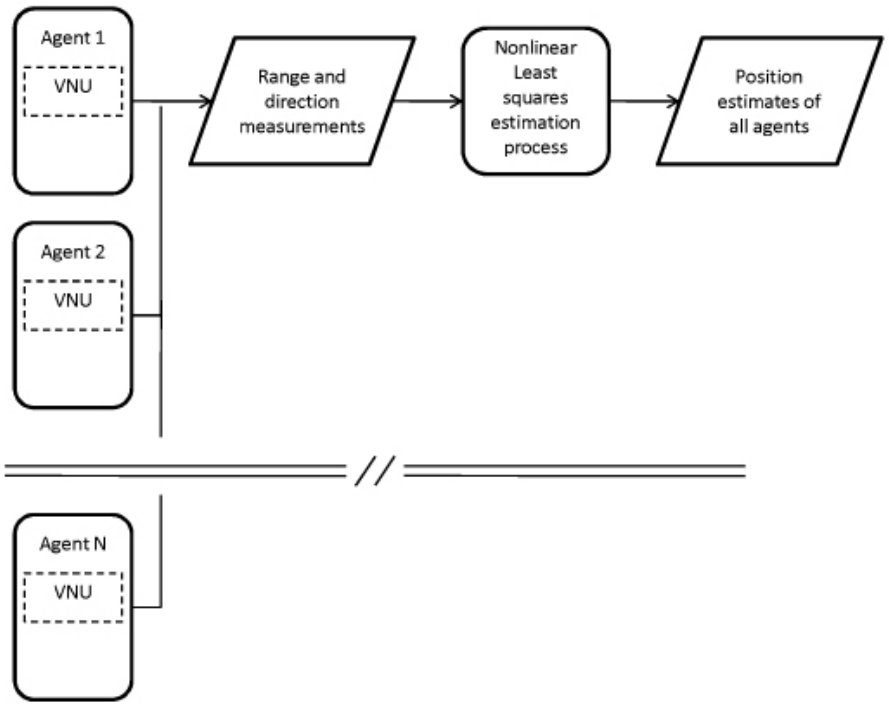


Fig. 9.2: Block diagram of the NLS vision system operation. The Visual Navigation Unit (VNU), present on each agent, provides range and direction measurements ($\widetilde{d}_{ij}, \widetilde{\alpha}_{ij}, \widetilde{\beta}_{ij}$), which is combined to provide position estimates for all of the agents.

It is clear that the use of the noisy coordinate estimates produced using this method is unsuitable for any form of precision control, as would be required within implementations such as interferometry. From the obtainable coordinate system it is desired to extract optimal estimations of the agent kinematic states within the supplied coordinate system: this is solved through the application of a Kalman filter.

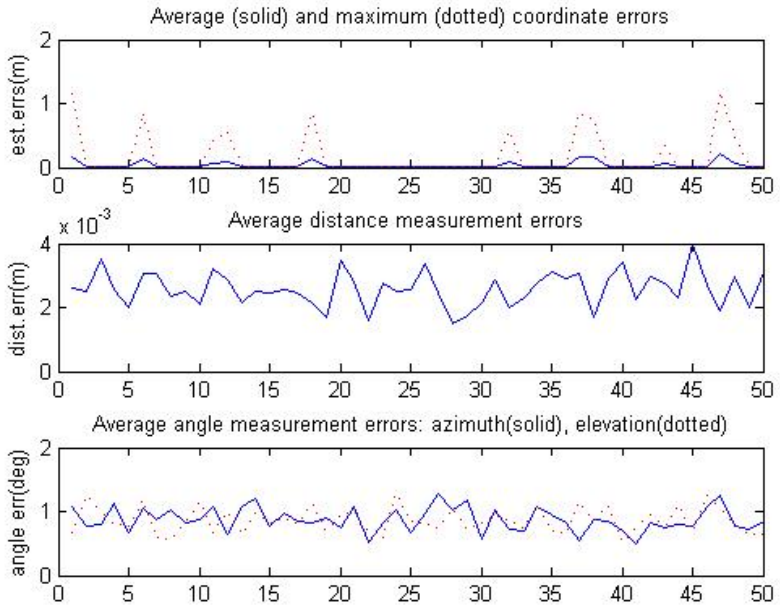


Fig. 9.3: Illustrative average and maximum errors of position and co-ordinate estimates under elevation and azimuth bounds of 1.5^o and distance measurement error bound of 5mm over 100 independent repetitions with randomly selected agent locations within a 100m cube.

9.3 Kalman Filtering

Kalman filters provide a means of extracting useful state information from a discrete stochastic process[4] by a recursive filtering process. Kalman filters operate by propagating the mean and covariance[5] of the state through time: the mean of the state is the Kalman filter estimate of the state and the covariance of the state is the covariance of the Kalman filter state estimate. Numerous forms of Kalman filters exist, from the 'simple' Kalman filter originally conceived by R.E. Kalman in 1958, which was extended by S.F. Schmidt and R.H. Battin prior to inclusion within the Apollo onboard guidance system, to the information, particle and a variety of square-root filters developed by numerous authors [137]. Rigorous mathematical derivations of the Kalman filter are widely available within

[4] A stochastic process is one whose behavior is non-deterministic in that a system's subsequent state is determined both by the process's predictable actions and by a random element.

[5] Covariance is a topic of probability theory and statics indicating a measure of how two variable change together.

literature (recommended texts include [137] and [138]) and an outline of the derivation, following the notation used within [138], will be provided here for completeness.

Given the linear discrete-time system

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1} \tag{9.4}$$

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k \tag{9.5}$$

where **F**, **G**, **H**, **w** and **v** represent the state transition model applied to the previous state $\mathbf{x}_{k-1}$, the control input model applied to the previous control input $\mathbf{u}_{k-1}$, the observation model which maps the true state space into the observed space and the two noise processes **w** and **v** which are white, zero mean, uncorrelated and have know covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ respectively.

We wish to estimate the state $\mathbf{x}_k$ based upon knowledge of the system dynamics and availability of noisy measurements $\mathbf{y}_k$. The manner in which this state estimate is formed depends upon the information available, since the method involves determining both *a priori* and *a posteriori* state estimates for the same state. The *a priori* state estimate is based upon all preceding measurements, up to but not including that at time k: this estimate will be denoted by $\widehat{\mathbf{x}}_k^+$. Once the measurement is made for time k, this information can be taken into account and form the *a posteriori* state estimate, denoted by $\widehat{\mathbf{x}}_k^-$. Naturally one would expect the *a posteriori* state estimate to be a better estimate.

For the subsequent discussion we will assume that the process is starting at time $k = 0$, that $'+'$ superscripts indicate estimations made *a priori* to the current measurement and $'-'$ superscripts indicate estimations made *a posteriori*. Furthermore, $\hat{\mathbf{x}}$ will be used to denote the estimated value of **x**, which is the true state and $E(h)$ indicates the 'expected' value of h.

The initial measurement is taken at time $k = 1$ and since at this time there are no previous measurements of **x**, $\hat{\mathbf{x}}^+$ can be defined as the expected value of the initial state $\mathbf{x}_0$:

$$\hat{\mathbf{x}}_0^+ = E(\mathbf{x}_0) \tag{9.6}$$

Denoting the covariance of the estimation error using $\mathbf{P}_k$ then we can define the *a priori* and *a posteriori* estimation error covariances as:

$$\mathbf{P}_k^- = E\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)^T\right] \tag{9.7}$$

$$\mathbf{P}_k^+ = E\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)(\mathbf{x}_k - \hat{\mathbf{x}}_k^+)^T\right] \tag{9.8}$$

It is now desired to compute $\hat{\mathbf{x}}_1^-$ given that we have $\hat{\mathbf{x}}_0^+$ which is complete through knowledge of Equation 9.4 and result in what is known as the time update equation for $\hat{\mathbf{x}}$:

$$\hat{\mathbf{x}}_k^- = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1}^+ + \mathbf{G}_{k-1}\mathbf{u}_{k-1} \tag{9.9}$$

This is followed by updating the state estimation error covariance, $\mathbf{P}$. We commence with $\mathbf{P}_0^+$, the covariance of the initial estimate $\hat{\mathbf{x}}_0$: if the initial state is known perfectly, then $\mathbf{P}_0^+ = 0$; if unknown then $\mathbf{P}_0^+ = \infty \cdot \mathbf{I}$. In general $\mathbf{P}_0^+$ represents the uncertainty of the initial $\hat{\mathbf{x}}_0$ estimate. The covariance of the state of a linear discrete time system propagates with time according to $\mathbf{P}_k = \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}$ and so for the system in question here, we can form the time update equation for $\mathbf{P}$ as:

$$\mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \tag{9.10}$$

Having formed the time update equations, we now need to form the measurement update equations for $\hat{\mathbf{x}}$ and $\mathbf{P}$: this will provide us with $\hat{\mathbf{x}}_k^+$ and $\mathbf{P}_k^+$. The availability of a measurement, $\mathbf{y}_k$, changes the estimate of $\mathbf{x}$ as follows:

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{P}_k^-\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1} && (9.11)\\
&= \mathbf{P}_k^+\mathbf{H}_k^T\mathbf{R}_k^{-1} && (9.12)\\
\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right) && (9.13)\\
\mathbf{P}_k^+ &= \left(I - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-\left(I - \mathbf{K}_k\mathbf{H}_k\right)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T && (9.14)\\
&= \left(I - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^- && (9.15)
\end{aligned}$$

where the notation used has been retained and $\mathbf{K}$ is referred to as the Kalman filter gain. The quantity $\left(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right)$ is known as the *innovation*: this is the part of the measurement which contains new information about the state.

Using these Kalman filter equations applied to the noisy coordinate data supplied by the vision system presented in the preceding Section, it is possible to filter the data to retrieve useful measurements which can be used within control methods. The application of this will be shown in the following Section.

9.4 Kalman Filtering For Position And Velocity

Application of the Kalman filter equations to noisy vision based coordinate data[6], as generated using the nonlinear least squares estimation process presented at the start of this Chapter, and based upon disturbances forces with a maximum bound of $0.1N$ acting on the double integrator dynamical model, was completed for two separate scenarios; each based upon three agents. The first scenario was from the provision of vision only data;

[6] Vision data was simulated subject to white, zero mean and uncorrelated noise of $\varepsilon_{ij} = 0.001^o$, $\eta_{ij} = 0.0011^o$ and $d_{ij} = 0.02m$

the second was based upon both vision and noisy accelerometer data[7]. A visualization of the typical motion for the three agents during the Kalman filter application is given within Figure 9.4

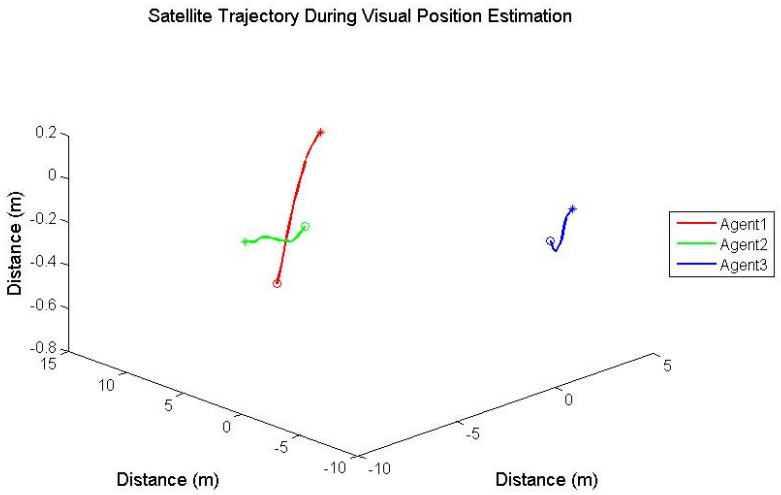


Fig. 9.4: Motion of spacecraft agents during state estimation process. Initial and final locations are indicated by asterisk and circular symbols respectively: a line trace denoting the path of the agent connects the two points.

Figures 9.5 and 9.6 show the state estimation errors for the spacecraft agents during and shortly after the Kalman filter tuning process. Inclusion of accelerometer data within the estate estimation process aids in both reducing and smoothing the state estimate.

[7] Accelerometer characteristics were based upon the O'Navi MEMS inertial measurement unit [139].

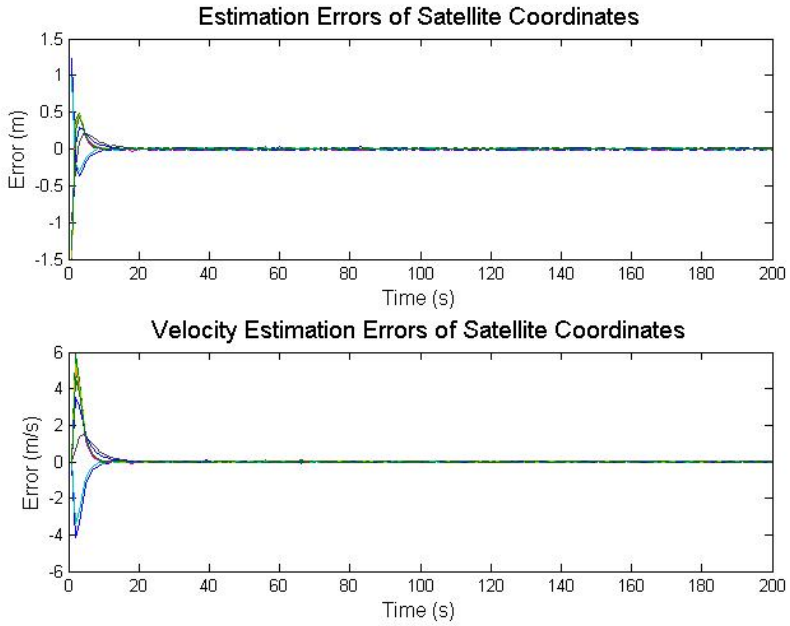


Fig. 9.5: Evolution of state estimation errors during Kalman filter application: implementation is based upon vision estimates only.

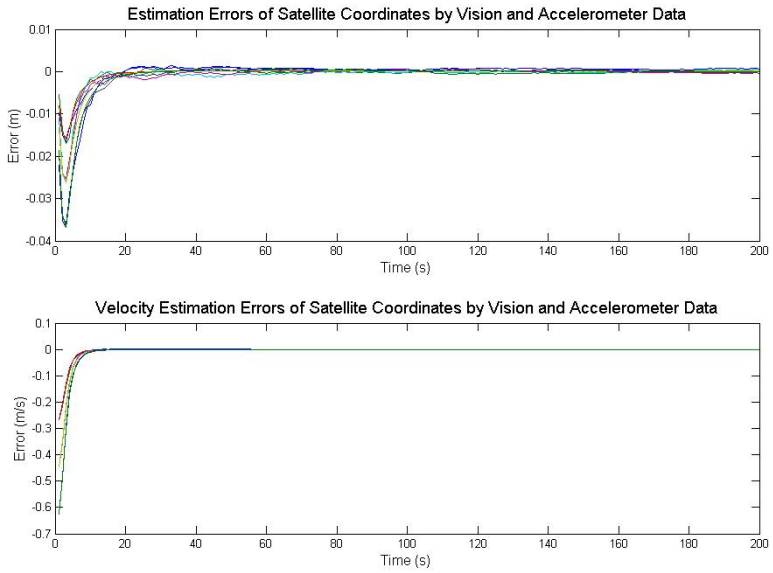


Fig. 9.6: Evolution of state estimation errors during Kalman filter application: implementation is based upon vision estimates and accelerometer data

9.5 Kalman Filtering For Attitude

A suitable Kalman filter for the estimation of position and velocity has been presented. It is also desired to obtain an estimate of the agent's current attitude and angular velocity so that this knowledge is available for latter control implementation.

In considering a traditionally equipped satellite, the presence of both rate integrating gyros and some form of device to obtain the current attitude[8] can be assumed; thus yielding high frequency provision of body mounted angular velocities and lower frequency provision of agent attitude. It is usual to propagate attitude knowledge using the strapdown equations presented within (7.7) via the measured angular body rates and the most recent attitude measurement; however it is well known that the output from a rate integrating gyro is subject to a random walk process leading to a measurement bias, in addition to general measurement noise. An extended Kalman filter is used as a solution to this problem by using the filter to determine the gyro bias in addition to a correction for the quaternion estimate: such a process has been well documented within [141, 142, 143, 144]. Of these papers, only the author of [142] and [143] presents a generic extended Kalman filter for implementation with a known quaternion measurement: the other papers are aimed towards process noise applied to direction vector measurements of known stars. None of these provide a complete presentation of the mathematics required to implement a discrete time computational method; moreover the derivation provided within [142, 143] is both incomplete and contains mistakes within the state equations used in the development of the filter for the update equations. The highly cited paper [141] also contains a sign mistake within the state equation for the body mounted state error equation. For these reasons a thorough derivation will be provided here.

9.5.1 The Lefferts/Markley/Shuster (LMS) Filter

The LMS filter is presented within [141] and is based upon defining the attitude first by the current attitude estimate and then by a small error relative to the body frame: it is this error which is to be determined and applied by the filter. In keeping with the discussion presented within Chapter 7 and the notation used for developing the linear Kalman filter[9], this can be expressed as

$$\mathbf{q}_{s/c} = \hat{\mathbf{q}} \otimes \mathbf{q}_e \tag{9.16}$$

[8] Here we are generalizing the availability of attitude information which may be obtained and prescribed globally through star sensors or locally through active vision techniques applied to camera observations [140]

[9] In keeping with the development of the linear Kalman filter, estimated values will be denoted by the "hat" symbol.

with $\hat{\mathbf{q}}$ representing the quaternion estimate and $\mathbf{q}_e$ representing the error quaternion, which can be expressed as:

$$\mathbf{q}_e = \mathbf{q}_{s/c} \otimes \hat{\mathbf{q}}^{-1} \tag{9.17}$$

The rate of update based on vision sensors is typically slow[10] and so between measurements attitude knowledge is propagated through use of rate integrating gyroscopes which are capable of significantly greater sampling rates[11]. As has previously been mentioned, rate integrating gyroscopes are subject to a random walk noise process leading to a measurement bias. This bias is also to be estimated by the filter to enable accurate attitude estimation.

Information is passed to the LMS filter in the form of noisy angular velocity measurements and noisy quaternion estimates, provided by the rate integrating gyroscopes and vision system respectively. The gyroscope measures both the true angular velocity, plus the noise introduced from bias and process:

$$\boldsymbol{\omega}^G_m = \boldsymbol{\omega}^G_{gyro} + \mathbf{b}^G_{gyro} + \boldsymbol{\eta}^G_{gyro} \tag{9.18}$$

where $\boldsymbol{\eta}^G_{gyro}$ is the gyroscope process noise and $\mathbf{b}^G_{gyro}$ is the gyroscope bias[12].

As presented within [142], the estimated body frame angular velocity can be constructed from gyroscope measurements as:

$$\hat{\boldsymbol{\omega}}^B_{s/c} = \mathbf{C}^T_{G/B}\left(\boldsymbol{\omega}^G_m\right) \tag{9.19a}$$

$$= \boldsymbol{\omega}^B_{s/c} - \mathbf{C}^T_{G/B}\left(\hat{\mathbf{b}}^G_{gyro} - \mathbf{b}^G_{gyro} + \boldsymbol{\eta}^G_{gyro}\right) \tag{9.19b}$$

$$= \boldsymbol{\omega}^B_{s/c} - \mathbf{C}^T_{G/B}\left(\Delta\mathbf{b}^G_{gyro} + \boldsymbol{\eta}^G_{gyro}\right) \tag{9.19c}$$

where $\mathbf{C}^T_{G/B}$ is the gyroscope to body direction cosine matrix and $\boldsymbol{\omega}^B$ is the angular velocity in the body frame. The obvious observation to be made here is that alignment of the gyroscope and agent body axes will reduce (9.19a) to the simpler form of $\hat{\boldsymbol{\omega}}^B_{s/c} = \boldsymbol{\omega}^G_m - \mathbf{b}^G_{gyro} - \boldsymbol{\eta}^G_{gyro}$.

Regarding the vision based quaternion measurements, computationally we can construct the noisy quaternion input, corrupted by white gaussian noise $\boldsymbol{\eta}_{vision}$ as

$$\mathbf{q}_{s/c} = \mathbf{q} \otimes \{\boldsymbol{\eta}_{vision}, 1\} \tag{9.20}$$

[10] Within [123] the update period for the ROSAT precision star tracker is given as one second

[11] Sampling rates for typical rate integrating gyroscopes are given within [123] as 0.04s (25Hz)

[12] The gyroscope bias is modeled as a random walk process, such that $\dot{\mathbf{b}} = \boldsymbol{\eta}_{bias}$

within which $\boldsymbol{\eta}_{vision}$ is a 3-vector. Within [142] a further step is made regarding the quaternion error measurement, where the measured error is formed through a combination of (9.17) and (9.20) to form:

$$\mathbf{q}_e = \mathbf{q} + \mathbf{C}^T_{C/B}\boldsymbol{\eta}_{vision} \tag{9.21}$$

where $\mathbf{C}^T_{C/B}$ represents the direction cosine matrix of the camera frame relative to the body frame. Within the implementation of [142], the direction cosine matrix is used within the computation of the Kalman gain; however the manner in which it is implemented results in the formation of the identity matrix.

System State Vector

We wish to model and estimate values for the error quaternion and gyroscope bias hence, in state space form, our system state vector will be $[\mathbf{q}_e, \mathbf{b}_{gyro}]^\top$. The derivation of a state space model for the time rate of change of both the quaternion error and gyroscope bias shall proceed in a similar manner to [141] and [142] where we will commence by differentiating our quaternion error presented within (9.16) to yield

$$\dot{\mathbf{q}}_{s/c} = \hat{\mathbf{q}} \otimes \dot{\mathbf{q}}_e + \mathbf{q}_e \otimes \dot{\hat{\mathbf{q}}} \tag{9.22}$$

by application of

$$\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix} \omega \\ 0 \end{bmatrix} \otimes \mathbf{q}$$

we result in

$$\frac{1}{2}\begin{bmatrix} \omega_{s/c} \\ 0 \end{bmatrix} \otimes \mathbf{q}_{s/c} = \hat{\mathbf{q}} \otimes \dot{\mathbf{q}}_e + \mathbf{q}_e \otimes \left\{ \frac{1}{2}\begin{bmatrix} \hat{\omega} \\ 0 \end{bmatrix} \otimes \hat{\mathbf{q}} \right\} \tag{9.23}$$

$$\dot{\mathbf{q}}_e = \frac{1}{2}\left\{ \begin{bmatrix} \omega_{s/c} \\ 0 \end{bmatrix} \otimes \mathbf{q}_{s/c} - \mathbf{q}_e \otimes \begin{bmatrix} \hat{\omega} \\ 0 \end{bmatrix} \right\} \tag{9.24}$$

which is in confirmation with [141]. Substitution of our gyroscope angular velocity measure (9.19a) results in

$$\dot{\mathbf{q}}_e = \frac{1}{2}\left\{ \begin{bmatrix} \hat{\omega} \\ 0 \end{bmatrix} \otimes \mathbf{q}_e - \mathbf{q}_e \otimes \begin{bmatrix} \hat{\omega} \\ 0 \end{bmatrix} \right\} - \frac{1}{2}\mathbf{C}^T_{G/B}\begin{bmatrix} \Delta\mathbf{b}_{gyro} + \boldsymbol{\eta}_{gyro} \\ 0 \end{bmatrix} \otimes \mathbf{q}_e \tag{9.25}$$

with application of (7.7) we can expand the quaternion cross product components to yield

$$\begin{aligned}
\dot{\mathbf{q}}_e &= \frac{1}{2}\left\{\begin{bmatrix} -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & \hat{\boldsymbol{\omega}} \\ -\hat{\boldsymbol{\omega}}^\top & 0 \end{bmatrix}\cdot\mathbf{q}_e - \begin{bmatrix} +\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & \hat{\boldsymbol{\omega}} \\ -\hat{\boldsymbol{\omega}}^\top & 0 \end{bmatrix}\cdot\mathbf{q}_e\right\} \\
&\quad -\frac{1}{2}\mathbf{C}_{G/B}^T\begin{bmatrix} \Delta\mathbf{b}_{gyro}+\boldsymbol{\eta}_{gyro} \\ 0 \end{bmatrix}\otimes\mathbf{q}_e && (9.26a)\\
&= \frac{1}{2}\begin{bmatrix} -2\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & 0 \\ 0 & 0 \end{bmatrix}\cdot\mathbf{q}_e \\
&\quad -\frac{1}{2}\mathbf{C}_{G/B}^T\begin{bmatrix} -\lfloor(\Delta\mathbf{b}_{gyro}+\boldsymbol{\eta}_{gyro})\times\rfloor & (\Delta\mathbf{b}_{gyro}+\boldsymbol{\eta}_{gyro}) \\ -(\Delta\mathbf{b}_{gyro}+\boldsymbol{\eta}_{gyro})^\top & 0 \end{bmatrix}\cdot\begin{bmatrix} \mathbf{q}_e \\ 1 \end{bmatrix} && (9.26b)\\
&= \begin{bmatrix} -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & 0 \\ 0 & 0 \end{bmatrix}\cdot\mathbf{q}_e - \frac{1}{2}\mathbf{C}_{G/B}^T\begin{bmatrix} (\Delta\mathbf{b}_{gyro}+\boldsymbol{\eta}_{gyro}) \\ 0 \end{bmatrix} + O^2 && (9.26c)\\
&= \begin{bmatrix} -\hat{\boldsymbol{\omega}}\times\mathbf{q}_e - \frac{1}{2}\mathbf{C}_{G/B}^T(\Delta\mathbf{b}_{gyro}+\boldsymbol{\eta}_{gyro}) \\ 0 \end{bmatrix} && (9.26d)
\end{aligned}$$

Based upon the equations relating to the gyroscope bias, we can directly write:

$$\Delta\dot{\mathbf{b}}_{gyro} = \eta_{bias} \tag{9.27}$$

Under the assumption that the error quaternion is small, we can construct the complete state space equation as:

$$\begin{aligned}
\frac{d}{dt}\begin{bmatrix} \bar{\mathbf{q}}_e \\ \Delta\mathbf{b}_{gyro} \end{bmatrix} &= \begin{bmatrix} -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & -\frac{1}{2}\mathbf{C}_{G/B}^T \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix}\begin{bmatrix} \bar{\mathbf{q}}_e \\ \Delta\mathbf{b}_{gyro} \end{bmatrix} \\
&\quad + \begin{bmatrix} -\frac{1}{2}\mathbf{C}_{G/B}^T & 0_{(3,3)} \\ 0_{(3,3)} & I_{(3,3)} \end{bmatrix}\begin{bmatrix} \boldsymbol{\eta}_{gyro} \\ \boldsymbol{\eta}_{bias} \end{bmatrix} && (9.28a)\\
\dot{\bar{\mathbf{x}}} &= \mathbf{F}_c\bar{\mathbf{x}} + \mathbf{G}_c\boldsymbol{\eta} && (9.28b)
\end{aligned}$$

where the c subscripts have been used to highlight the fact that the system and noise matrices are based upon a continuous time formulation. It is important to note that the quaternion component has been reduced to represent only the vector part, represented by $\bar{\mathbf{q}}$, since through the assumption of $\mathbf{q}_e$ being small the scalar component will remain at unity. It is at this point where our system model diverges from that presented within [141, 142, 143], which contain sign errors within the $\mathbf{F}_c$ matrix and do not all consider misaligned gyroscopes.

Time Discretisation

In order to implement the derived error propagation model presented within (9.28a), it is required to discretise the model, in addition to forming the state transition matrix and

system noise covariance denoted by $\mathbf{\Phi}$ and $\mathbf{Q}_d$ respectively. Discretisation of continuous time systems is well documented [145, 146], formally deriving the discrete time state transition matrix of a continuous time system as:

$$\begin{aligned} \mathbf{\Phi}(t+\Delta t) &= exp(\mathbf{F}_c \Delta t) && \text{(9.29a)} \\ &= \mathbf{I}_{(6,6)} + \mathbf{F}_c \Delta t + \frac{1}{2!}\mathbf{F}_c^2 \Delta t^2 + \ldots && \text{(9.29b)} \end{aligned}$$

Following the development of [146] and [147], we can expand the powers of the F_c matrix to yield:

$$\begin{aligned} F_c &= \begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}} \times \rfloor & -\frac{1}{2}\mathbf{C}_{G/B}^T \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix} \\ F_c^2 &= \begin{bmatrix} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 & \frac{1}{2}\mathbf{C}_{G/B}^T \lfloor \hat{\boldsymbol{\omega}} \times \rfloor \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix} \\ F_c^3 &= \begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}} \times \rfloor^3 & -\frac{1}{2}\mathbf{C}_{G/B}^T \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix} \\ F_c^4 &= \begin{bmatrix} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^4 & \frac{1}{2}\mathbf{C}_{G/B}^T \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^3 \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix} \\ F_c^5 &= \begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}} \times \rfloor^5 & -\frac{1}{2}\mathbf{C}_{G/B}^T \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^4 \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix} \\ &\vdots \\ F_c^n &= \begin{bmatrix} \pm\lfloor \hat{\boldsymbol{\omega}} \times \rfloor^n & \pm\frac{1}{2}\mathbf{C}_{G/B}^T \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^{(n-1)} \\ 0_{(3,3)} & 0_{(3,3)} \end{bmatrix} \end{aligned}$$

where the $\pm$ component is $+$ for even n and negative otherwise. Combining the powers of F_c within (9.29b) we are presented with the discrete time state transition matrix being of the form

$$F_d = \begin{bmatrix} \Theta & \Psi \\ 0_{(3,3)} & I_{(3,3)} \end{bmatrix} \tag{9.30}$$

We will now consider the development of Θ and Ψ individually.

$$\Theta = I_{(3,3)} - \lfloor \hat{\boldsymbol{\omega}} \times \rfloor \Delta t + \frac{1}{2!} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \Delta t^2 - \frac{1}{3!} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^3 \Delta t^3 + \frac{1}{4!} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^4 \Delta t^4 - \ldots \quad (9.31)$$

By employing the properties of skew symmetric matrices Θ can be reformulated into:

$$\begin{aligned} \Theta &= I_{(3,3)} - \frac{1}{|\hat{\omega}|} \left(|\hat{\omega}| \Delta t - \frac{1}{3!} |\hat{\omega}|^3 \Delta t^3 + \ldots \right) \lfloor \hat{\boldsymbol{\omega}} \times \rfloor \\ &\quad + \frac{1}{|\hat{\omega}|^2} \left(1 - \left\{ 1 - \frac{1}{2!} |\hat{\omega}|^2 \Delta t + \frac{1}{4!} |\hat{\omega}|^4 \Delta t - \ldots \right\} \right) \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \end{aligned} \quad (9.32)$$

and through recognition of the series expansion for $sin(x)$ and $cos(x)$, we can simplify further to:

$$\Theta = I_{(3,3)} - \frac{1}{|\hat{\omega}|} sin(|\hat{\omega}| \Delta t) \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + \frac{1}{|\hat{\omega}|^2} \left(1 - cos(|\boldsymbol{\omega}| \Delta t)\right) \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \quad (9.33)$$

clearly due to the division by $|\hat{\omega}|$, small values of ω will lead to numerical instability. By taking the limit and applying L'Hopital's rule, we result in:

$$\boldsymbol{\Theta} = \mathbf{I}_{(3,3)} - \Delta t \lfloor \hat{\omega} \times \rfloor + \frac{\Delta t^2}{2} \lfloor \hat{\omega} \times \rfloor^2 \quad (9.34)$$

Repeating the same procedure for the Ψ matrix, we find that

$$\begin{aligned} \Psi &= \frac{\mathbf{C}_{G/B}^T}{2} \left[-I_{(3,3)} \Delta t + \frac{\Delta t^2}{2!} \lfloor \hat{\omega} \times \rfloor - \frac{\Delta t^3}{3!} \lfloor \hat{\omega} \times \rfloor^2 + \frac{\Delta t^4}{4!} \lfloor \hat{\omega} \times \rfloor^3 - \ldots \right] \\ &= \frac{\mathbf{C}_{G/B}^T}{2} \left[-I_{(3,3)} \Delta t + \left(\frac{\Delta t^2}{2!} - \frac{\Delta t^4}{4!} |\hat{\omega}|^2 + \ldots \right) \lfloor \hat{\omega} \times \rfloor + \left(-\frac{\Delta t^3}{3!} + \frac{\Delta t^5}{5!} |\hat{\omega}|^2 - \ldots \right) \lfloor \hat{\omega} \times \rfloor^2 \right] \\ &= \frac{\mathbf{C}_{G/B}^T}{2} \left[-I_{(3,3)} \Delta t + \frac{1}{|\hat{\omega}|} \left(1 - cos(|\hat{\omega}| \Delta t)\right) \lfloor \hat{\omega} \times \rfloor - \frac{1}{|\hat{\omega}|} \left(|\hat{\omega}| \Delta t - sin(|\hat{\omega}| \Delta t) \right) \lfloor \hat{\omega} \times \rfloor^2 \right] \\ &= \frac{\mathbf{C}_{G/B}^T}{2} \left[-I_{(3,3)} \Delta t + \frac{\Delta t^2}{2} \lfloor \hat{\omega} \times \rfloor - \frac{\Delta t^3}{6} \lfloor \hat{\omega} \times \rfloor^2 \right] \end{aligned} \quad (9.35)$$

from which we are now in the position to construct the discrete time state transition matrix initially presented within (9.30).

Determination of the noise covariance within the discrete time system is detailed with [146], where the required computation is given as:

$$\mathbf{Q}_d = \int_{t_k}^{t_k+1} \Phi(t_{k+1}, \tau) \mathbf{G}_c \mathbf{Q}_c^\top(\tau) \Phi^\top(t_{k+1}, \tau) . d\tau \quad (9.36)$$

which for our system model is

$$\mathbf{Q}_d = \int_{t_k}^{t_k+1} \begin{bmatrix} \frac{1}{4}\sigma_1^2 I_{(3,3)} + \sigma_2^2 \Psi\Psi^\top & \sigma_2^2\Psi \\ \sigma_2^2\Psi^\top & \sigma_2^2 I_{(3,3)} \end{bmatrix} d\tau \tag{9.37}$$

$$= \begin{bmatrix} \mathbf{Q}_{(11)} & \mathbf{Q}_{(12)} \\ \mathbf{Q}_{(12)}^\top & \mathbf{Q}_{(22)} \end{bmatrix} \tag{9.38}$$

within the formulation of which, use is made of the fact that since $\mathbf{C}_{G/B}^T$ and Θ are rotation matrices, $\mathbf{C}_{G/B}^T \cdot \mathbf{C}_{G/B}^{T\top}$ and $\Theta \cdot \Theta^\top$ are identity matrices. In following with the procedures completed in [147], who deal with a similar but not identical matrix, the elements of (9.38) are determined as:

$$\mathbf{Q}_{(11)} = \frac{\sigma_1^2 \Delta t I_{(3,3)}}{4} + \frac{\sigma_2^2}{4}\left[I_{(3,3)}\frac{\Delta t^3}{3} + \frac{1}{|\hat{\omega}|^5}\left(\frac{|\hat{\omega}|^3\Delta t^3}{3} + 2sin(|\hat{\omega}|\Delta t) - 2|\hat{\omega}|\Delta t \right) \lfloor \hat{\omega}\times \rfloor^2 \right] \tag{9.39}$$

$$\mathbf{Q}_{(12)} = \frac{\sigma_1^2 \mathbf{C}_{G/B}^T}{2}\left[-\frac{I_{(3,3)}\Delta t^2}{2} + \frac{1}{|\hat{\omega}|^3}\left(|\hat{\omega}|\Delta t - sin(|\hat{\omega}|\Delta t) \right) \lfloor \hat{\omega}\times \rfloor \right]$$
$$- \frac{\sigma_1^2 \mathbf{C}_{G/B}^T}{2}\left[\frac{1}{|\hat{\omega}|^4}\left(\frac{|\hat{\omega}|^2\Delta t^2}{2} + cos(|\hat{\omega}|\Delta t) - 1 \right) \lfloor \hat{\omega}\times \rfloor^2 \right] \tag{9.40}$$

$$\mathbf{Q}_{(22)} = \sigma_2^2 \Delta t I_{(3,3)} \tag{9.41}$$

which after application of series expansion and L'Hôpital's rule, reduce to:

$$\mathbf{Q}_{(11)} = \frac{\sigma_1^2 \Delta t I_{(3,3)}}{4} + \frac{\sigma_2^2}{4}\left(I_{(3,3)}\frac{\Delta t^3}{3} + \frac{2\Delta t^5}{5!} \lfloor \hat{\omega}\times \rfloor^2 \right) \tag{9.42}$$

$$\mathbf{Q}_{(12)} = \frac{\sigma_1^2 \mathbf{C}_{G/B}^T}{2}\left(-\frac{I_{(3,3)}\Delta t^2}{2} + \frac{\Delta t^3}{3!} \lfloor \hat{\omega}\times \rfloor - \frac{\Delta t^4}{4!} \lfloor \hat{\omega}\times \rfloor^2 \right) \tag{9.43}$$

$$\mathbf{Q}_{(22)} = \sigma_2^2 \Delta t I_{(3,3)} \tag{9.44}$$

Integration of the quaternion can be completed by solving the first order differential equation given within (7.7), though it is desired to use a closed form solution for application within discrete time instances. A zeroth and first order closed form quaternion integrator is derived within [147], which are completed under the assumption of constant and linear varying ω respectively. These are both given below:

$$\mathbf{q}^{0^{th}}(t_{k+1}) = \left[I_{(4,4)} + \frac{\Delta t}{2}\Omega(\omega) \right] \otimes \mathbf{q}(t_k) \tag{9.45}$$

$$\mathbf{q}^{1^{st}}(t_{k+1}) = \left[exp\left(\frac{1}{2}\Omega(\bar{\omega})\Delta t \right) + \frac{\Delta t^2}{48}\left\{ \Omega(\omega^+)\Omega(\omega) - \Omega(\omega)\Omega(\omega^+) \right\} \right] \mathbf{q}(t_k) \tag{9.46}$$

within which $\bar{\omega} = \frac{\omega(t_k)+\omega(t_{k+1})}{2}$, $\omega = \omega(t_k)$, $\omega^+ = \omega(t_{k+1})$ and the $\Omega(\omega)$ operand retains the formulation presented within (7.7).

Implementation

We are now in the position to implement the standard Extended Kalman Filter algorithm in order to estimate the system state. There are two separate instances of propagation relating to the availability of data at the gyroscope and vision measurement frequencies.

For the high frequency estimation and propagation, which is completed by gyroscopic measurements, the following procedure is followed:

1. Bias propagation is completed under the assumption of a constant bias estimate, hence $\hat{\mathbf{b}}^{+} = \hat{\mathbf{b}}^{-}$.

2. Through the current gyroscope measurement, $\omega_{gyro}(t_k)$, and the current bias estimate, $\hat{\mathbf{b}}$, the current angular velocity estimate is formed through (9.19a) within which we use $\omega = \omega_{gyro} - \hat{\mathbf{b}}$.

3. The current quaternion is propagated in discrete time using the equation listed for either a zeroth or first order integrator, (9.45) or (9.46) respectively.

4. The state transition matrix and discrete time noise covariance matrix are calculated using equations (9.29b) through (9.44).

5. The state covariance matrix is computed using the standard EKF equation.

$$\mathbf{P}^{+} = \Phi\mathbf{P}^{-}\Phi^{\top} + \mathbf{Q}_d \tag{9.47}$$

Once high accuracy vision based quaternion estimates are available, an alternative update process is completed within which the quaternion estimate is refined and a new estimate for the gyroscope bias is made. We are assuming the availability of a new (but noisy) quaternion measurement as given by (9.20), and so the measurement matrix is given simply as

$$\mathbf{H} = \begin{bmatrix}\mathbf{I}_{(3,3)} & \mathbf{0}_{(3,3)}\end{bmatrix} \tag{9.48}$$

and this is used within the following procedure:

1. Compute the Kalman gain via

$$\mathbf{K} = \mathbf{P}^{-}\mathbf{H}^{\top}(\mathbf{H}\mathbf{P}^{-}\mathbf{H}^{\top} + \mathbf{R}^{\top})^{-1} \tag{9.49}$$

within which $\mathbf{R}$ is the $(3,3)$ measurement noise covariance matrix.

2. Calculate the quaternion residual, which in this application is the error between the propagated quaternion and the measured quaternion using

$$\delta \mathbf{q}_e = \mathbf{q}^-_{s/c} \otimes \mathbf{q}^{-1}_{measured} \tag{9.50}$$

3. Compute the required correction to be made, which we shall denote by $\delta \bar{\mathbf{x}}$, using the current Kalman gain and residual:

$$\delta \bar{\mathbf{x}} = \begin{bmatrix} \hat{\bar{\mathbf{q}}}^+_e \\ \Delta \hat{b}^+ \end{bmatrix} \tag{9.51}$$

$$= \mathbf{K} \delta \bar{\mathbf{q}}_e \tag{9.52}$$

4. Conditionally format the new quaternion error estimate $\hat{\mathbf{q}}^+_e$, to ensure that the quaternion unity constraint is not violated[13], based upon the product $\hat{\bar{\mathbf{q}}}^{+\top}_e \hat{\bar{\mathbf{q}}}^+_e$. If the product $\hat{\bar{\mathbf{q}}}^{+\top}_e \hat{\bar{\mathbf{q}}}^+_e \leq 1$

$$\hat{\mathbf{q}}^+_e = \begin{bmatrix} \hat{\bar{\mathbf{q}}}^+_e \\ \sqrt{1 + \hat{\bar{\mathbf{q}}}^{+\top}_e \hat{\bar{\mathbf{q}}}^+_e} \end{bmatrix} \tag{9.53}$$

Otherwise, if $\hat{\bar{\mathbf{q}}}^{+\top}_e \hat{\bar{\mathbf{q}}}^+_e > 1$

$$\hat{\mathbf{q}}^+_e = \frac{1}{\sqrt{1 + \hat{\bar{\mathbf{q}}}^{+\top}_e \hat{\bar{\mathbf{q}}}^+_e}} \begin{bmatrix} \hat{\bar{\mathbf{q}}}^+_e \\ 1 \end{bmatrix} \tag{9.54}$$

5. Update the estimated agent quaternion by application of (9.16).

6. Update the estimated gyroscope bias using:

$$\hat{\mathbf{b}}^+ = \hat{\mathbf{b}}^- + \Delta \hat{\mathbf{b}}^+ \tag{9.55}$$

7. Update the covariance matrix

$$\mathbf{P}^+ = \mathbf{P}^- - \mathbf{K}\mathbf{H}\mathbf{P}^- \tag{9.56}$$

[13] Within the EKF formulation we have developed a model based upon the assumption that the error quaternion is small and hence only the vector part of the quaternion is of interest and the scalar part remains at unity. In application this assumption may lead to violation of the quaternion constraint that $|\mathbf{q}| = \sqrt{\mathbf{q}^\top \mathbf{q}} \equiv 1$. In order to circumvent this possibility, the EKF determined quaternion error is conditionally converted into a full 4-vector quaternion.

Filter Performance

Implementation of the presented extended Kalman filter for attitude estimation, using gyroscopic propagation and low frequency update with accurate vision based quaternion information, based upon the performance characteristics given within [123], provided increased accuracy within state estimation.

Figures 9.7, 9.8 and 9.9 display the resulting output during the simulate EKF implementation. Figure 9.7 shows the estimation process for the gyroscopic bias (a result of drift), with the actual bias depicted with a red line and the estimated value depicted using a black line. Figure 9.8 shows the error between the true angular velocity and those given by direct measurement form the gyro and estimated rotation rates from the filter, depicted by the red and black lines respectively: it is evident that the estimated rotation rate corresponds with greater accuracy to the true rotation rate. Figure 9.9 shows the error between the true components of the quaternion vector compared to estimates based upon gyroscopic propagation[14] and those from the filter; here gyroscopically propagated values are represented by a red and that of the filter estimate with a black line. It is observed that quaternion estimation based upon gyroscopic propagation quickly diverges from the true value, whilst the quaternion propagated using the EKF is difficult to discern from zero.

[14] Gyroscopic propagation was completed using the strap-down equations presented in Chapter 7

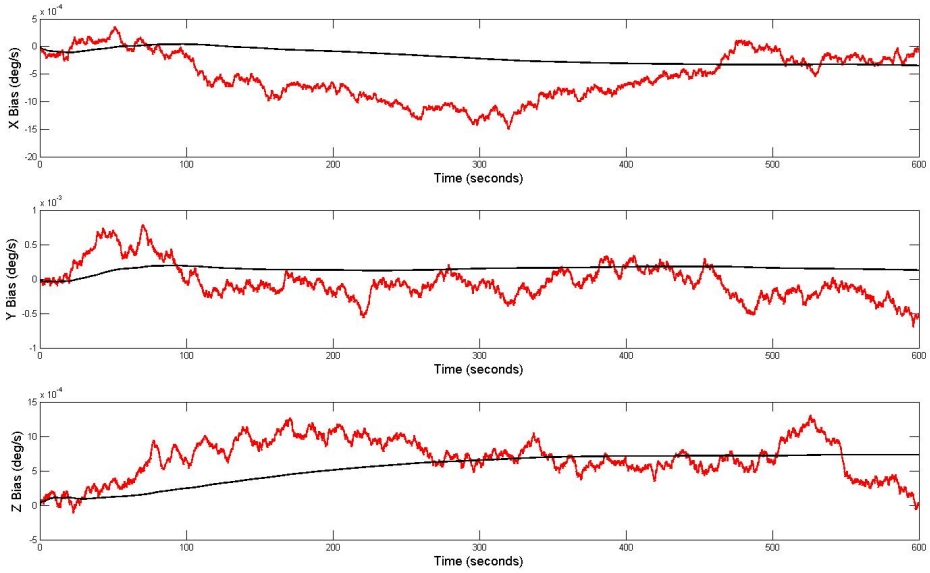


Fig. 9.7: Evolution of gyroscope bias (red) and estimated bias (black) during EKF implementation based upon a random walk process with a maximum drift rate of 1deg/hr.

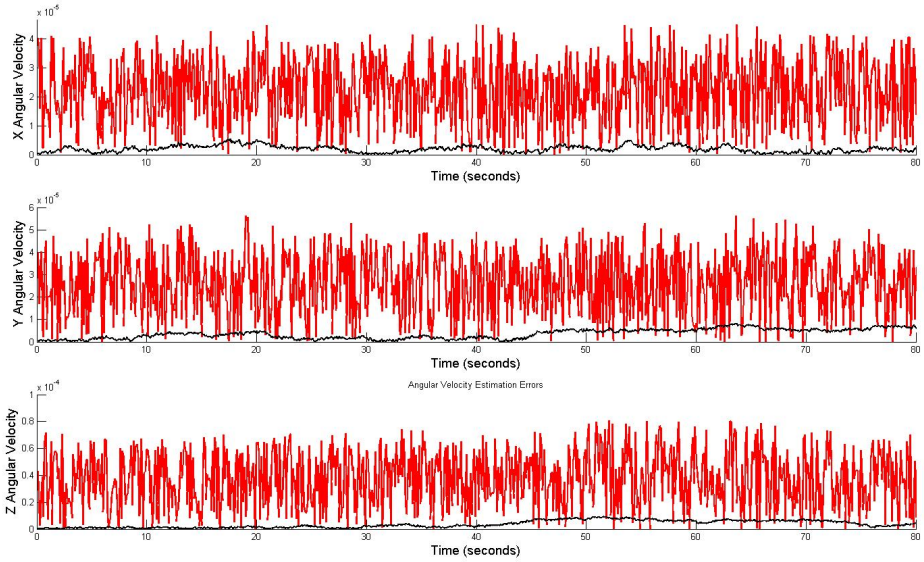


Fig. 9.8: Rotation rate error between the true rotation rate and those taken directly from the gyroscope and those estimated by the filter (red and black respectively) during EKF implementation: note that the estimated values correspond to the true value with greater accuracy.

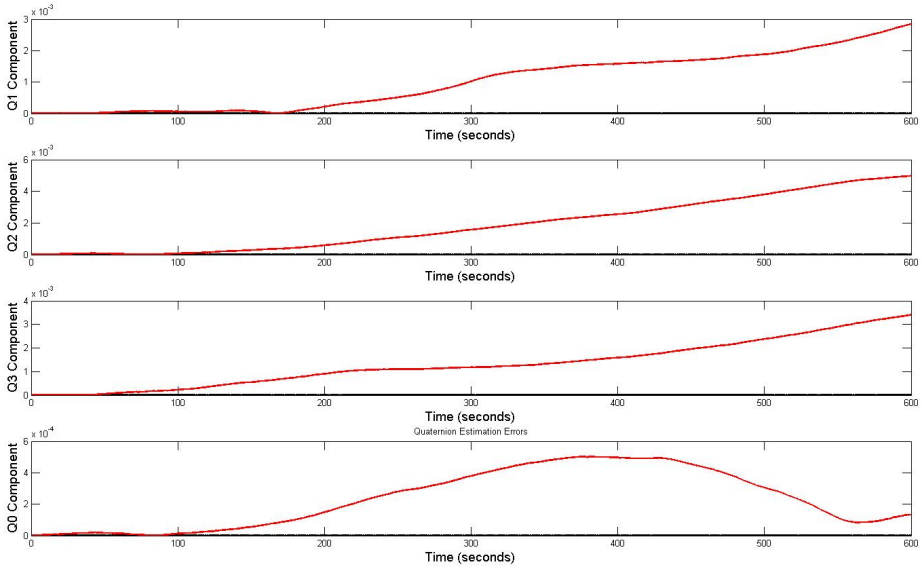


Fig. 9.9: Quaternion component error between the true components of the quaternion vector compared to estimates based upon gyroscopic propagation and those from the filter. Gyroscopically propagated values are represented by a red and that of the filter estimate with a black line.

9.6 Combined Filter

Agent state estimation for both translation *and* rotation states can be achieved through a hybrid state estimation filter composed of both filters which have been presented individually. A schematic for the construction of the hybrid filter is shown within Figure 9.10, which shows how both presented filters can be used to obtain complete state estimation. Within the Figure, the acronyms NLS, LKF and EKF are for the Nonlinear Least Squares position estimate, the Linear Kalman Filter and Extended Kalman Filter, respectively. The Visual Navigation Unit (VNU) and Inertial Measurement Unit (IMU) are the sensors used to provide state estimation data. Whilst it is true that a single filter could have been designed to perform state estimation based upon all vision information, it was chosen to separate the two vision processes during the filter implementation as these processes are likely to have differing update rates depending on the complexity of the computations to be completed: this is especially pertinent when considering the method used to determine the relative positions, which requires centralized pre-processing of state information prior to being used within a linear Kalman filter.

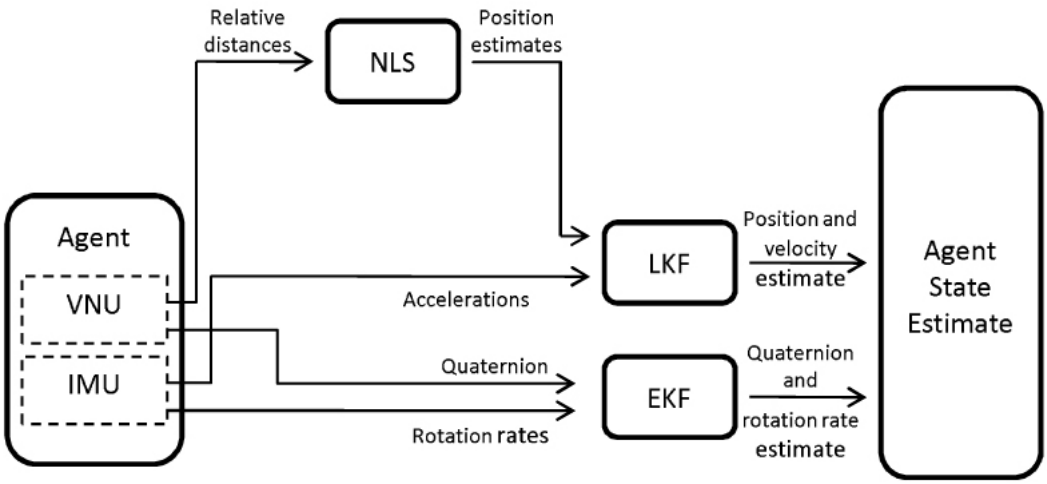


Fig. 9.10: Schematic for a hybrid filter to provide both translational and rotational state information, for a single agent, using the filters presented previously. The acronyms NLS, LKF and EKF are for the Nonlinear Least Squares position estimate, the Linear Kalman Filter and Extended Kalman Filter, respectively. The Visual Navigation Unit (VNU) and Inertial Measurement Unit (IMU) are the sensors used to provide state estimation data.

9.7 The Cluster State Estimation problem

In the previous Sections, state estimation of a single agent has been discussed. In this Section a description of how to join these estimates, to form a complete cluster dynamical state, will be given.

Agent positions and velocities within the cluster are given expressly by the presented NLS and LKF methods presented: here a single agent is tasked with handling all range, azimuth and elevation data from the agents within the cluster to form the position state estimates for the entire cluster.

Velocity and rotational state (both orientation and rotation rate) estimates are the responsibility of the individual agents within the cluster: each agent is tasked with determining their own attitude quaternion and rotation rate within the designated frame of reference using the presented EKF.

Upon each state estimation iteration, the individual agent state estimates may be combined to form a complete agent cluster state estimate. Note that in practice each agent only requires their complete state within the cluster in addition to position estimates of each other agent within the cluster. Figure 9.11 depicts the cluster state estimation process. Within the Figure, the circular dashed component is used to represent the NLS output data, which is shared among the agent community.

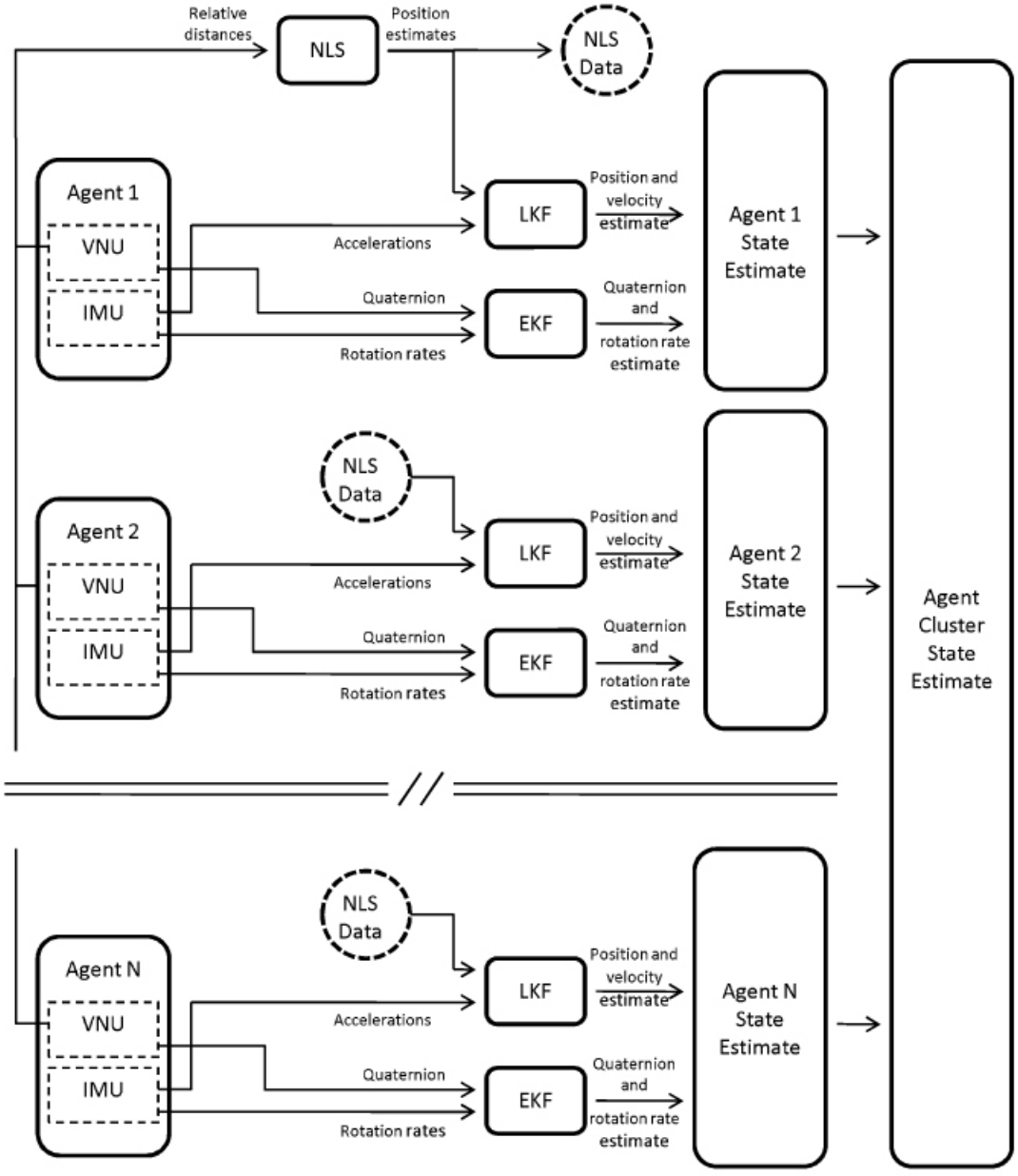


Fig. 9.11: Schematic for a hybrid filter to provide both translational and rotational state information, for the entire agent cluster, using the filters presented previously. The acronyms NLS, LKF and EKF are for the Nonlinear Least Squares position estimate, the Linear Kalman Filter and Extended Kalman Filter, respectively. The Visual Navigation Unit (VNU) and Inertial Measurement Unit (IMU) are the sensors used to provide state estimation data.

9.8 Chapter Summary

This Chapter has presented a vision based methodology for the full state estimation of a multi-agent system operating within the space environment. Two separate filters, for translational and rotational states, were developed and integrated to obtain full state estimation from noisy measurements. The translational state estimation filter, applied to a satellite cluster operating in a low Earth orbit, was published within [148].

Simulation of the state estimation process was completed using MATLAB (release 2006b) and we are consequently presented with the common phrase "simulation is doomed to succeed". Whilst the simulations were performed in the presence of noise representative of the individual components, it should be noted that the simulation results presented should only be considered as *representative* of possible system performance rather than an absolute proof of system functionality. Nevertheless, through the resulting simulation output, it is apparent that the derived state estimation methods are suitable and certainly capable of performing the desired function: estimation of $6DoF$ state from noisy measurements.

The accuracy of the entire system is obviously directly related to the accuracy of measurement devices implemented onboard the agent system. It is unrealistic to expect nano-meter precision from devices accurate to cm level, however, using the data fusion and filtering processes, we are capable of obtaining state estimation to a greater precision than that of a single device.

10. SPACECRAFT AGENT CONTROL ACTION

This Chapter will focus upon the formulation of a control law, to be implemented within discrete time and applied to the interferometer agent system, for the purpose of controlling agent position and attitude. The control law will make use of the potential function guidance method and Kalman filtered state estimation processes detailed in Chapters 8 and 9 respectively.

The control method to be implemented is that of sliding mode control (SMC), a nonlinear control method which first arose within the context of variable structure systems; SMC is now the leading method to design for this system classification [149]. There is a wealth of literature concerning SMC within continuous time systems, the fundamental analysis and design methods dating as far back as the early 1930's [150].

10.1 SMC Within Continuous Time

Sliding mode control within continuous time systems proceeds with two primary stages, those of manifold design and sliding condition design.

Manifold design Selection of the sliding mode hyper-surface or manifold, denoted by $\sigma(t)$, is completed in such a manner that when the system is maintained to this manifold (at $\sigma(t) \triangleq 0$), the desired behavior is exhibited. The terminology 'manifold', 'sliding surface', 'switching surface', 'hyper-surface' and 'sliding mode' all relate to the same entity and are used interchangeably within the literature.

Sliding condition design This stage involves determination of the required feedback gains such that the system trajectory intersects and remains on the prescribed manifold. This process is traditionally completed by defining an *equivalent control*[1] term in conjunction with a *switching control*[2] term, the combination of which is proven to be asymptotically stable using Lyapunov theory.

[1] The equivalent control term is designed such that upon intersection with the manifold at $\sigma(t) \triangleq 0$, motion is constrained to the surface

[2] The switching term is designed such as to guarantee convergence to the sliding manifold and has a theoretically infinite switching frequency.

Although in the preceding summary a single manifold has been mentioned, the reader should not infer that only a single manifold is ever implemented within SMC: indeed multiple manifolds may be used within a controller design depending on the degrees of freedom to be controlled. This leads to various control methodologies in which differing methods are used to deal with the sliding surfaces. Hierarchical sliding mode control is employed by driving a single manifold to the intersection within finite time and then progressing sequentially through the set of sliding modes, whilst maintaining preceding modes, until $\sum \sigma(t)_i \triangleq 0$, $i = 1,2,...,n$. Due to the nature of this approach, hierarchical sliding mode control is also referred to as a fixed order switching scheme [151][152]. Such a method is limited in that the prescribed order of reaching sliding modes may not be ideal and can result in high control effort. In contrast, a 'free-order' sliding mode process operates on a first-reach-first-switch scheme in which the order of sliding modes is not specified. Although possessing more desirable dynamical characteristics, by definition no direct control is available over sliding modes: this can be disadvantageous in cases when a precedence is required due to actuation limitations[3].

Within implementation, the switching control term may lead to chattering: finite frequency, finite amplitude oscillations appearing as a result of the high frequency switching of the sliding mode controller exciting unmodeled dynamics. This chattering is not the controller switching action itself, since in the ideal case the switching frequency tends to infinity. One of the more popular methods of addressing this occurrence is via the 'boundary layer solution', which seeks to eliminate control discontinuities and switching action within the control loop. This is achieved by introduction of a saturation function replacing the sign function internal to the switching control term and has the form:

$$\text{sat}(y) = \begin{cases} y & \text{if } |y| \leqslant 1 \\ sgn(y) & \text{otherwise} \end{cases}$$

Whilst it is true that no 'real' sliding motion takes place with this adjustment, due to the switching action being modified by a continuous approximation, sliding mode control methodologies are exploited in the design of the controller and consequently numerous authors retain the classification of SMC for such systems.

10.2 SMC Within Discrete Time

Although consideration within the idealized continuous time world leads to elegant and mathematically rigorous solutions, when considering practical problems, controllers are

[3] Here actuation limitations may equally refer to power limitations as to force output

implemented within a discrete time environment as a consequence of digital microprocessors [145][149]. An immediate consideration here is violation of the theoretically infinite switching frequency because the switching will be limited to the sampling rate and may lead to discretization chatter[4]. As we will see, chatter will not be significant within 6 DOF spacecraft motion control.

An increasing amount of literature is appearing regarding the development of sliding mode control schemes within a discrete time environment, the fundamental aspects being within publication [149]. The design methodology for discrete time sliding mode control (DTSMC) is in essence identical to that for a continuous time counter part: manifold design followed by sliding condition design. The first stage, that of manifold design, is identical to that of the continuous time case, aside from the implicit construct of the sliding surface being such that we are forming $\sigma(k)$ where k represents the discrete time instance[5]. However the definition of a reaching law is not as simplistic. Despite the huge breadth of literature concerning the development of DTSMC, the quantification of what represents system motion for 'sliding modes in discrete time' and the necessary reaching conditions to achieve this is a highly contended issue within the control community [153, 154, 155, 156]. Direct abstractions from continuous time implementation yield requirements such as the somewhat obvious requirement of $|\sigma(k+1)| < |\sigma(k)|$ with the desire to achieve $|\sigma(k+1)| = 0$, as presented within [153, 154]. A more complex set of requirements to entail sliding motion in discrete time is presented by [155], within which it is stated that for discrete sliding motion to occur, the $\sigma(k) = 0$ boundary must be crossed and crossed infinitely often thereafter whilst remaining within a prescribed region of the manifold. Such requirements are very strict and reside firmly within the purist SMC theorist group, relating back to the original requirements of infinite switching upon manifold intersection. Within all practical applications of sliding mode control, continuous or otherwise, methods are sought to eliminate the chattering associated with crossing of the $\sigma = 0$ boundary as opposed to seeking this instance. Subsequent analysis within this thesis will not take the requirements proposed within [155] as necessary, though the former requirement of $|\sigma(k+1)| < |\sigma(k)|$ will be assumed as both necessary and sufficient.

[4] Discretization chatter is a separate phenomenon to that of the previously introduced chatter resultant from unmodeled dynamics.

[5] As computer controlled systems can only act at quantized time instants it is assumed that a series state estimates and control actions happen at a sequence of time instances $t_1, t_2, t_3, ..., t_\infty$ that for simplicity are indexed by $k = 1, 2, 3, ..., \infty$. In general sampling can be periodic with fixed sampling period, or with some small variations in $h_k = t_{k+1} - t_k$.

10.3 Controller Development

This Section will present a discrete time controller developed for the proposed multi-agent system. Although the presentation here will relate to a generalized agent, the controller will be formulated in such a way that the control law can be applied universally throughout the agent system.

Within Chapter 9, methods were presented such that estimation of the complete agent state, within a local coordinate system and denoted by $\hat{\mathbf{x}}_i(k)$, was possible. Consequently the complete agent state can be assumed available and comprised of:

- Relative position vector estimates for agent i to the MS, based upon a coordinate system attached to the MS. Denoted as $\hat{\mathbf{d}}_i(k)$.
- Relative position vector estimates for agent i to all other agents j, where $j = 1,2,3,...,(N-1)$ based upon a coordinate system attached to agent i. Denoted as $\hat{\mathbf{s}}_{ij}(k)$. Note that it is these values which are used within the Kalman filter to obtain the preceding $\hat{\mathbf{d}}_i(k)$ estimate.
- Agent orientation estimates relative to the global coordinate system, given in the quaternion notation and denoted $\hat{\mathbf{q}}_i(k)$ within which the notation of (7.5) is retained.
- Rates of change of agent position and orientation vector estimates, denoted by $\hat{\mathbf{v}}_i(k)$ and $\hat{\omega}_i(k)$ respectively.
- Agent state estimates for the next time instance, through the internal Kalman filter *a priori* prediction mechanism, $\hat{\mathbf{x}}_i(k^+)$.

It is these entities which shall be used to construct the discrete time error states for implementation within a digitized control system. Relating back to Chapter 5, within which the multi-agent interferometer mission is presented, the requirements of full state regulation must be considered in combination within inter-agent collision avoidance. The coordinate system adopted, that of a local system attached to a lead agent, in combination with regulation to a fixed configuration indicates that we are seeking solution to a point tracking problem, as presented within Chapter 8, that is augmented to account for collision avoidance.

As introduced within Chapter 8, collision avoidance is accounted for by introducing an inter-agent avoidance potential between agent i and all other agents j, where $j = 1,2,3,...,(N-1)$, and for our system this can be written as:

$$\Upsilon_i(k) = \sum_{j=1}^{N-1} \hat{\mathbf{s}}_{ij}(k) exp\left(-\frac{\|\hat{\mathbf{s}}_{ij}(k)\|}{r_j}\right), \tag{10.1}$$

In this notation r_j is an additional weighting parameter dictating the particular sphere of influence about agent j. The desired response is motion down the negative gradient of (10.1): in our context the potential function can be viewed as a method of developing a kinematic field, with the production of a desired velocity vector to satisfy agent collision avoidance.

At this stage it will be assumed that the relevant decision process to assign agents to array point locations has been completed and that each agent has knowledge of its destined location, given in the coordinate frame developed within Kalman filter mechanism presented within Chapter 9. This array position will be denoted by $\mathbf{d}_i^d$, where the superscript of d is used to express *desired* and the i subscript retains the usual meaning such that $\mathbf{d}^d$ is relating to agent i. In addition to the positional requirement provided by $\mathbf{d}_i^d$, a particular attitude is to be obtained and this is given using the quaternion notation as $\mathbf{q}_i^d$ and it is assumed that the agent also has knowledge of the desired quaternion[6]. Both $\mathbf{d}_i^d$ and $\mathbf{q}_i^d$ are combined with their derivatives to provide a complete nominal state. Complete agent guidance is provided by a combination of agent errors from this nominal state and the collision avoidance criterion provided by (10.1).

The agent errors are comprised of positional error from the required location in 3-space, translational velocity error between agent i and the MS to ensure accurate tracking and augmented by the avoidance criterion (10.1), orientation error and angular velocity error to ensure accurate global orientation. Although in principle full state feedback is implied, by use of Kalman filtering principles provided within Chapter 9, full state feedback can be achieved without the need of measuring relative translational velocities: these can be generated internally using only relative positional information.

Using the superscript of e to express *error* components, we can explicitly define the relative agent errors to be used in controller action. For position and velocity we have:

$$\mathbf{d}_i^e(k) = \mathbf{\Psi}\mathbf{d}_i^d(k) - \hat{\mathbf{d}}_i(k) \tag{10.2}$$

$$\mathbf{v}_i^e(k) = c_v\mathbf{\Psi}\cdot\mathbf{\Upsilon}_i(k) - \hat{\mathbf{v}}_i(k) \tag{10.3}$$

in which $\mathbf{\Psi}$ represents an inertial to body conversion matrix and c_v represents a scalar gain factor to scale the avoidance potential function.

For agent orientation error, denoted by $\mathbf{q}_i^e$, the quaternion notation as presented within Chapter 7 will be used, as will the formulation of the error quaternion given by $\mathbf{q}_i^e = \mathbf{q}_i^{d^{-1}} \otimes \hat{\mathbf{q}}_i$. A minimal representation of the agent quaternion error for agent i, permitting

[6] The desired quaternion may be given as an attitude relative to the celestial sphere, in which case star sensor information will be used directly, or as a relative attitude to neighboring agents which may be obtained using visual processing techniques such as the POSIT algorithm [140].

representation as a 3-vector, is given by:

$$\bar{\mathbf{q}}_i^e(k) = sgn(q_{i_4}^e(k)) \cdot \begin{bmatrix} q_1^e(k) \\ q_2^e(k) \\ q_3^e(k) \end{bmatrix} \tag{10.4}$$

In a somewhat more straight forward manner, the angular speed vector error for agent i is defined by:

$$\omega_i^e(k) = \omega_i^d(k) - \hat{\omega}_i(k) \tag{10.5}$$

Within the discussion relating to DTSMC, it was presented that the initial step in the formulation of such a controller was that of formulating the sliding surfaces, σ_i, such that upon reaching $\sigma_i = 0$, the desired system motion is exhibited. For our consideration, where both position and attitude are to be controlled in combination with translational and rotational velocities, the necessity for two sliding surfaces is implied. The elements to these sliding surfaces for our agent control scheme have already been presented and are the error states given within (10.2, 10.3, 10.4 & 10.5). By combining the relevant elements we construct our required sliding surfaces for each agent, i, as:

$$\sigma_i^1(k) = \mathbf{d}_i^e(k) + \mathbf{v}_i^e(k) \tag{10.6}$$
$$\sigma_i^2(k) = \omega_i^e(k) + \bar{\mathbf{q}}_i^e(k) \tag{10.7}$$

For notational purposes, the two sliding surfaces may be shown more concisely as:

$$\begin{aligned} \Sigma_i(k) &= \begin{bmatrix} \sigma_i^1(k) \\ \sigma_i^2(k) \end{bmatrix} \\ &= \mathbf{G}_i(k)\mathbf{x}_i^e(k) \end{aligned} \tag{10.8}$$

where

$$\begin{aligned} \mathbf{G}_i(k) &= \begin{bmatrix} I_{(3,3)} & I_{(3,3)} & 0_{(3,3)} & 0_{(3,1)} & 0_{(3,3)} \\ 0_{(3,3)} & 0_{(3,3)} & sgn\{q_4(k)\}I_{(3,3)} & 0_{(3,1)} & I_{(3,3)} \end{bmatrix} \\ \mathbf{x}_i^e(k) &= \begin{bmatrix} \mathbf{d}_i^e(k) & \mathbf{v}_i^e(k) & \mathbf{q}_i^e(k) & \omega_i^e(k) \end{bmatrix}^T \end{aligned}$$

Looking into the structure of the sliding surfaces, it is possible to gain an insight into the stability of the system under the constraint of the prescribed surfaces for both translational and attitude motion. Considering the agent translational surface within (10.6) and

prescribing $\sigma_i^1(k) = 0$, by use of our previously defined agent errors (10.2 & 10.3) and the kinematics given in (7.13) we obtain

$$\dot{\mathbf{d}}_i^e(k) = -(I + \mathbf{\Omega}(\hat{\boldsymbol{\omega}}, k))\hat{\mathbf{d}}_i^{kf}(k) - \mathbf{\Psi}\mathbf{d}_i^d(k) - k_v \cdot \mathbf{\Upsilon}_i(\mathbf{s}_{ij}, k) \tag{10.9}$$

Since the first term within (10.9) is bounded and the remaining two terms are stable, input-output stability can be guaranteed.

Now considering the agent attitude motion constrained by $\sigma_i^2(k)$, again setting $\sigma_i^2(k) = 0$ results in $\omega_i^e(k) = -\bar{\mathbf{q}}_i^e(k)$. Combining this with the quaternion kinematic equation (7.7) and dropping the (k) notation for clarity, we obtain

$$\begin{aligned} \dot{\mathbf{q}}_i^e &= \frac{1}{2}\mathbf{\Omega}|_{(-\bar{\mathbf{q}}_i^e)} \cdot \mathbf{q}_i^e \\ &= -\frac{1}{2} sgn\left(\mathbf{q}_{i_4}^e\right) \mathbf{\Omega}|_{[q_{i_1}^e, q_{i_2}^e, q_{i_3}^e]} \cdot \mathbf{q}_i^e \end{aligned}$$

which can be divided further into

$$\dot{\mathbf{q}}_{i_4}^e = \frac{1}{2}\left[1 - \mathbf{q}_{i_4}^{e\,2}\right] sgn\left(\mathbf{q}_{i_4}^e\right) \tag{10.10}$$

$$\dot{\mathbf{q}}_{i_n}^e = -\frac{1}{2}\left[\mathbf{q}_{i_4}^e \cdot sgn\left(\mathbf{q}_{i_4}^e\right)\right] \mathbf{q}_{i_n}^e \tag{10.11}$$

$$n = 1, 2, 3.$$

Using (10.10 & 10.11) it can be proven that when on the sliding surface, the error quaternion progresses to $[0, 0, 0, sgn(\mathbf{q}_{i_4}^e)]$, which is zero attitude error by definition. It now remains to complete the controller through the sliding condition design such that the desired motion is enforced.

10.3.1 Reaching Law Approach

Within Section 10.2 the condition taken for the occurrence of sliding motion was given as $|\sigma(k+1)| < |\sigma(k)|$. Such a requirement is referred to as a 'reaching condition', which gives rise to reaching law approach for control system design [155]. This approach involves the formulation of a suitable reaching law, which directly specifies the dynamics of the switching function and from this a variable structure control (VSC) law is synthesized. Such a method has the advantage of prescribing the dynamic characteristics of the reaching mode[7].

[7] A reaching law is defined as a differential equation which specifies the dynamics of a switching function $\Sigma(x)$. Note that the differential equation of an asymptotically stable $\Sigma(x)$ is itself a reaching condition.

Various continuous time reaching law methods are presented within [155]. We shall make use of a discrete time version of a power rate reaching law, abstracted from the continuous time counterpart and given as:

$$\Sigma(k+1)-\Sigma(k) \quad = \quad -\varepsilon h|\Sigma(k)|^{\varsigma} sgn\{\Sigma(k)\} \tag{10.12}$$

where $0<\varsigma<1$, ε is a positive definite scalar constant and h represents the sampling period.

Looking at our compound sliding surface given within (10.8), and considering an incremental progression of the surface, $\Sigma_i(k+1)-\Sigma_i(k)$, we obtain the difference between the surfaces. Both $\Sigma_i(k)$ and $\Sigma_i(k+1)$ can be estimated by Kalman filtering methods, using the values of $\hat{\mathbf{x}}_i(k)$ and $\hat{\mathbf{x}}_i(k^+)$ respectively. For small enough sampling times (below the Nyquist sampling rate) the difference in these sliding surfaces can be considered to be the estimated differential of the sliding surface for the given time instance k, which shall be denoted as $\dot{\hat{\Sigma}}_i(k)$. Noting that this estimated sliding surface differential is based upon unforced dynamics, we must include the influence of control forces to our sliding surface equation, which relates to the central term within (7.15c) and is given by $\triangle\Sigma_i(k) = h(\mathbf{G}_i(k)\mathbf{B})\mathbf{u}_i(k)$. The controlled dynamics of the sliding surface are therefore given as:

$$\dot{\Sigma}_i(k) \quad = \quad \dot{\hat{\Sigma}}_i(k)+\triangle\Sigma_i(k) \tag{10.13}$$

We seek to enforce a specific dynamic upon this sliding surface differential, $\dot{\Sigma}_i(k)$, such that it follows the power rate reaching law given within (10.12). Using this we can directly formulate our desired control response as:

$$\mathbf{u}_i^*(k) \quad = \quad (h\mathbf{G}_i(k)\mathbf{B})^{-1}\left(-\dot{\hat{\Sigma}}_i(k)-\varepsilon h|\Sigma_i(k)|^{\varsigma} sgn\{\Sigma_i(k)\}\right) \tag{10.14}$$

Where $\mathbf{G}_i(k)\mathbf{B}$ is a square invertible matrix and the *star* superscript is used to indicate that the control signal is an ideal response. To provide stabilizing control the upper bound of the available control resources, denoted as u_0, must satisfy the following inequality:

$$u_0 \quad \geq \quad (\mathbf{G}_i(k)\mathbf{B})^{-1}\cdot\left(|\mathbf{G}_i(k)\mathbf{f}_d(k)|+|\mathbf{G}_i(k)Cu_d(k)|+\frac{\delta}{h}\right) \tag{10.15}$$

Proof. Examining the sliding surfaces presented within (10.8), the sliding surface one time step ahead can be predicted using (7.15c) as

$$\Sigma_i(k+1) \quad = \quad \Sigma_i(k)+h\mathbf{G}_i(k)\left[\mathbf{f}_d(k)+Bu_i^*(k)+Cu_d(k)\right]$$

for which

$$\dot{\mathbf{\Sigma}}_i(k) \simeq \mathbf{\Sigma}_i(k+1) - \mathbf{\Sigma}_i(k)$$

Taking a Lyapunov function candidate of $V(k) = \frac{1}{2}\Sigma_i^T(k)\Sigma_i(k)$, which has a derivative of $\dot{V}(k) = \Sigma_i(k)\dot{\Sigma}_i(k)$, we require that $\dot{V}(k) \leq -\delta|\Sigma_i(k)| \; \forall t$. By substitution,

$$\begin{aligned} \dot{V}(k) &= \Sigma_i(k)\mathbf{G}_i(k)h[f(\mathbf{x},k) + \mathbf{GBu}(k) + \mathbf{G}_i(k)\mathbf{Cu}_d(k)] \\ &\leq -\delta|\Sigma_i(k)| \end{aligned}$$

therefore

$$\mathbf{u}(k) \geq (\mathbf{G}_i(k)\mathbf{B})^{-1}\left(|\mathbf{G}_i(k)f(\mathbf{x},k)| + |\mathbf{G}_i(k)\mathbf{Cu}_d(k)| + \frac{\delta}{h}\right)$$

hence to permit controlled change of the defined sliding surfaces, the control resources must satisfy (10.15). □

In some instances the signal command may be greater than the available control resources will permit and for such circumstances, when $\|u_i^*(k)\| > u_0$, the control signal must be altered to enable the appropriate response. This is achieved by implementing the following control modification:

$$\mathbf{u}_i(k) = \begin{cases} \mathbf{u}_i^*(k) & \text{if } \|\mathbf{u}_i^*(k)\| \leq u_0 \\ u_0 \frac{\mathbf{u}_i^*(k)}{\|\mathbf{u}_i^*(k)\|} & \text{if } \|\mathbf{u}_i^*(k)\| > u_0 \end{cases} \tag{10.16}$$

It now remains to prove that upon the control modification provided by (10.16), a stabilizing control mode will be achieved. This is equivalent to the need of assuring that the time rate of change of the sliding surface remains negative upon the application of (10.16).

Proof. Considering our fundamental dynamic equation, (7.15c), and the definition of our sliding surfaces, (10.8), we may state:

$$\Sigma_i(k+1) = \Sigma_i(k) + h\mathbf{G}_i(k)\dot{x}(t)|_{t=kh} \tag{10.17a}$$

$$= \Sigma_i(k) + hG\left[f_d(k) + Bu_i^*(k) + Cu_d(k)\right] \tag{10.17b}$$

which under the condition of $\|u_i^*(k)\| > u_0$, can be expressed as:

$$\begin{aligned}\Sigma_i(k+1)-\Sigma_i(k) &= h\mathbf{G}_i(k)\left[f_d(k)+Cu_d(k)+Bu_0\frac{u_i^*(k)}{\|u_i^*(k)\|}\right] \\ &= h\mathbf{G}_i(k)[f_d(k)+Cu_d] \\ &\quad -\frac{u_o}{\|u_i^*(k)\|}\cdot\left[\dot{\hat{\Sigma}}_i(k)+\varepsilon h|\Sigma_i(k)|^{\varsigma}sgn\{\Sigma_i(k)\}\right]\end{aligned}$$

By use of (10.14) and (10.15) we obtain the result that

$$\Sigma_i(k+1)-\Sigma_i(k) \leq -\delta \tag{10.18}$$

Therefore $\mathbf{\Sigma}_i(k)$ remains negative and after a finite amount of steps, $\mathbf{u}_i^*(k)$ will be within the admissible domain of $\|\mathbf{u}_i^*(k)\| < u_0$, upon which the desired reaching rate law can be enforced. □

10.4 Chapter Summary

A discrete time sliding mode controller has been developed for the purposes of controlling agent motion for the interferometer scenario. Potential functions are integrated with errors based upon departure from a higher level decision process regarding position allocation in order to formulate the sliding surfaces. State information is provided through the Kalman filter methods presented within Chapter 9 and these quantities are used directly within the control formulation, which is based upon a discrete time exponential reaching rate.

The control strategy presented makes use of the internal model parameters encapsulated within the **B** matrix. These parameters can be updated online in order to refine the control process, thus implementing an adaptive control element within the scheme, as is completed within [148] and [149]. This would be advantageous when parameters such as the inertial matrix, **J**, are not know accurately or time varying; which would be the case when considering the usage of agent propellant resources.

The decision process relating to agent location was not entered upon within this Chapter and the output decision was assumed directly available to the agent. The following Chapter will present methods for formulating decisions within the agent community.

11. SPACECRAFT AGENT DECISION METHODS

Methods to obtain desired multi-agent actions were presented within Chapter 4, in the form of behavioral and decision making processes. This Chapter will expand upon agent action through decision, by application of CPDA and DPA methods[1], to metrics relevant for the decision problem at hand.

The primary decisions required of the spacecraft agent system, and those considered within this thesis, include:

- Initial dispersal configuration and respective AAP allocation to the set of possible optimal array locations as detailed within Chapter 3. The set of these optimal AAPs will be denoted as Λ_m, where m relates to the number of array points in the optimal configuration.
- Reconfiguration in the instance of agent failure to a new optimal array configuration with a reduced number of functional agents.
- Reconfiguration of the optimal array to maximize the mission duration.

The first two of these instances are subsets of an AAP allocation problem, but with differing preconditions; the final consideration relates to an optimization problem. Two methods to achieve the desired agent response will be considered: centralized planning with distributed execution (CPDE) and distributed planning and execution (DPE). CPDE represents a classical approach to the solution of the proposed problem through use of optimization methods; DPE is that which takes a multi-agent systems approach and here will be based upon the use of auction methods. All decisions will be based upon cost functions built from criteria pertinent to the PSSI application: consequently the cost functions involved within the decision processes will be conceptually identical, though used in a different manner. The cost function primitives to be used in formulating a decision will now be introduced.

[1] Referring back to Chapter 4 we are using the terminology of Centralized Planing with Distributed Action (CPDA) and Distributed Planning and Action (DPA).

11.1 Cost Function Primitives

In the instance of spacecraft motion, the most pertinent criterion is that to minimize fuel expenditure, since minimization of fuel usage permits the possibility of launch with a lower fuel mass fraction, or a longer duration mission with a fixed fuel mass fraction. As a result the cost function itself is composed of the ΔV[2] required to move to, and maintain, a particular location in space, in addition to the remaining fuel fraction for an agent being considered. Methods for determining these costing parameters will now be presented.

11.1.1 Position Acquisition ΔV

The required ΔV for an agent to move to a position can be estimated by using planning or predictive methods. The local dynamics between the mother-satellite and an agent, where the mother-satellite is constrained to a halo orbit and the agent is within the vicinity of the constrained mother-satellite, is given within (7.16) and repeated here for ease of continuation:

$$\delta\dot{\mathbf{x}}^w_{pw} = \mathbf{A}(\mathbf{x}^w_{lw}, t)\delta\mathbf{x}^w_{pw} \tag{11.1}$$

where $\mathbf{x}^w_{lw}$ relates to the mother-satellite and $\mathbf{x}^w_{pw}$ to a proximate agent, both in the world frame of reference. $\delta\mathbf{x}^w_{pw}$ can relate to the position of an agent from the mother-satellite, $\delta\mathbf{x}_i$, or the location of a desired position from the mother-satellite, $\delta\mathbf{x}_j$, where $\delta\mathbf{x}_j \in \Lambda_m$.

We are presented with two distinct options for forming a ΔV estimate using the dynamical model presented within (7.16), the initial relative agent state $\delta\mathbf{x}^w_{pw}$ and a desired relative agent state $\delta\mathbf{x}_j$: that of optimal path planning and motion simulation.

Optimal Path Planning Optimal path planning techniques involve the use of computational methods to form a path (or trajectory) from one dynamical state to another, such that the developed path is optimal with respect to selected criteria, which may be distance, time, fuel consumption or a combination thereof. For this problem, we are interested in finding the fuel optimal path from the initial agent state to the desired agent state which corresponds to an array location, within the dynamical regime specified by (7.16). Such a path optimization scheme, as presented by this author within [148], can be completed by the use of linear programming methods readily available within computational packages such as MATLAB or Mosek[3]. The

[2] ΔV is used to represent the required change in velocity for an agent and may also be considered synonymous to the propellant mass consumed.

[3] The MOSEK Optimization Software is designed to solve large-scale mathematical optimization problems. MOSEK provides specialized solvers for linear programming, mixed integer programming and many types of nonlinear convex optimization problems.

scheme presented within [148] does not take into account avoidance constraints: naturally we would desire that the resultant path does not pass through the mother-satellite. Such constraints can be inserted into the problem formulation such that we create a mixed integer programming problem (MIP), as presented within [157, 158]. Solutions to MIP problems are currently only solvable by specialist linear programming packages such as GLPK[4] and CPLEX[5], which may be interfaced via MATLAB using AMPL[6].

Using this method it is possible to minimize the fuel expenditure for agent transition between states subject to actuation, avoidance and time constraints. Whilst this is highly desirable, we are faced with the consideration that such optimization methods are incredibly expensive in terms of computation and in the presence of agent failure would require online re-computation: the reactive nature availed by MAS is lost. Moreover the developed agent skill set for control is that of point regulation and not path following, as would be required for execution of the optimal path generated. Whilst it is entirely possible to use optimal path planning methods to produce ΔV estimates relating to array location assignments and then use the controller developed within Chapter 10 as a separate instance upon completion of the agent decision process, this would not make complete use of the optimal path planning output and result in unnecessary computation.

Motion Simulation Through use of (7.16) as the dynamical system model and an error state of $\delta\mathbf{x}_i - \delta\mathbf{x}_j$, continuous time modeling can be used to determine the ΔV to reduce this error to zero. A continuous time sliding mode controller, inclusive of avoidance constraints by use of potential function methods and applicable to this problem is presented by this author within [163] and in sEnglish format within Appendix D. Through this method, the ΔV required for agent i to transition between states under the constraint of actuator bounds and avoidance requirements can rapidly be determined and thus yield the ΔV contribution $\Delta V_{i,j}^{move}$.

[4] The GLPK (GNU Linear Programming Kit) package is intended for solving large-scale linear programming, mixed integer programming, and other related problems. It is a set of routines written in ANSI C and organized in the form of a callable library[159].

[5] ILOG CPLEX (often informally referred to simply as CPLEX) is an optimization software package. It is named for the simplex method and the C programming language, although today it contains interior point methods and interfaces in the $C++$, $C\sharp$, and Java languages [160]

[6] AMPL is a comprehensive and powerful algebraic modeling language for the formulation of linear and nonlinear optimization problems, in discrete or continuous variables [161, 162].

11.1.2 Position Maintenance ΔV

Upon acquisition of a possible array-point, $\delta\mathbf{x}_j$, the ΔV associated with continual maintenance of this position must be evaluated for a given time period. Under the assumption that an agent has reached the desired location, the ΔV required to maintain this position may be determined via direct integration of (7.16) with $\delta\mathbf{x}_j$ fixed. This will produce an estimate for the ΔV cost of agent i to maintain position j for a specified time period and yield the contribution $\Delta V_{i,j}^{maintain}$.

An alternative method for determining the current position maintenance cost is obviously available directly from the agent: provided that the agent is capable of monitoring it's propellant reserves (and thus it's derivative) and/or estimate propellant usage based upon current thruster status. Through this method any errors present in the estimation of the $A(x,t)$ matrix is avoided, as we would be dealing directly with actual agent measurements during a given time period.

It should be noted that since $A(x,t)$ is dependent upon time, maintenance ΔV estimations extracted through either of the presented methods are only relevant for short term estimation, unless one is considering complete orbits.

11.1.3 Remaining Fuel Fraction

Consideration of the remaining fuel fraction (RFF) is required since the cost to a community for an agent with low propellant reserves to complete a task is greater than an alternative agent with high propellant reserves performing the same task. This is because the latter agent is less likely to deplete its fuel supply and thus end its operational life: agent communities suffer through the loss of community members and in this instance directly results in the reduction of interferometer performance.

Agents will consume propellant at different rates dependant upon their initial deployment, as a direct consequence of being maintained at varying distances from a nominal halo orbit and mathematically specified within our previous consideration for position maintenance ΔV. It is therefore prudent to use the remaining fuel fraction (RFF), for the agent concerned, to influence the resultant cost function.

11.2 AAP Allocation Decision

This Section will consider the decision instances of AAP allocation, which are required upon initial agent dispersal from the mother-satellite and upon detected total failure of an agent within the community, necessitating commodity redistribution to maintain an optimal array. Upon formulation of relevant cost functions to be used in the decision process,

themselves generated by the primitives described in the preceding Section, centralized or distributed processes may be used to form an appropriate decision relating to the problem encapsulated within the cost function. The cost functions relating to AAP allocation and the associated decision process will now be presented.

11.2.1 Cost Functions

There are two discrete instances necessitating the AAP allocation, and consequently we use two differing cost functions.

Initial Deployment Upon initial deployment of the agent system, the fuel status of each agent will be assumed identical and the consideration of position maintenance irrelevant. The pertinent factor is that relating to the change in state upon initial ejection from the mother-satellite to that relating to the respective array-point allocations. Consequently we are presented with a minimal cost function for agent i, based solely upon the position acquisition ΔV, as:

$$COST_{i,j}^{eject} = \Delta V_{i,j}^{move} \tag{11.2}$$

Redistribution Subsequent decision instances, when agents have been stationed at the optimal set of array points for a non-negligible amount of time, will result in a cost function influenced by differing fuel reserves within the agent community. Consequently all of the cost function primitives are pertinent to the cost function construction for agent i, which is given as:

$$COST_{i,j}^{redistribute} = (\Delta V_{i,j}^{move} + \Delta V_{i,j}^{maintain}) \cdot (RFF_i)^{-1} \tag{11.3}$$

11.2.2 Centralized Planning With Distributed Action: CPDA

In the instance of a centralized implementation it is the task of a central agent (nominally the mother-satellite) to consider the relevant cost functions associated with every possible AAP permutation. Using (11.2) or (11.3), depending upon the decision instance, it is possible to generate a cost matrix for the set of N agents and set of D desired positions resulting in a (N,D) matrix containing all possible instances and associated costs for N agents to move to D positions. Within the cost matrix, shown in (11.4), each column corresponds to a possible array-point an agent can take, from the set Λ_m, and the row entries within the column represent the associated cost for the agent to move to the position: $c_{(i,j)}$ is the cost of assigning agent i to desired position j.

$$\mathbf{CM} = \begin{bmatrix} c_{(1,1)} & c_{(1,2)} & \cdots & c_{(1,D)} \\ c_{(2,1)} & c_{(2,2)} & \cdots & c_{(2,D)} \\ \vdots & & \ddots & \vdots \\ c_{(N,1)} & c_{(N,2)} & \cdots & c_{(N,D)} \end{bmatrix} \tag{11.4}$$

An optimal centralized planning mechanism seeks to assign each agent to a position such that the total cost incurred is minimized. This is a linear assignment problem and requires the selection of N elements of **CM**, such that there is exactly one element in each row and one in each column selected and the sum of the corresponding costs is a minimum. Such an assignment problem is a classical problem within the field of linear programming solvable using the 'Hungarian algorithm', Genetic Algorithms or Simulated Annealing [164, 165].

In a completely centralized scheme it would be the responsibility of the mother-satellite to estimate the cost function primitives for all the AAP permutations, form and solve the cost matrix given within (11.4) and then relay the assignments to the respective agents; the agents themselves are responsible for autonomously achieving the array-point assignment. This method represents the application of CPDA, as presented within Chapter 4.

A more computationally efficient method would utilize each agent to compute their own cost functions relating to all possible array-point locations and relay this data-set to the central agent, wherein the cost matrix is constructed and solved. Such an implementation still holds to the method of CPDA, since the final assignment is chosen through a central method utilizing complete knowledge of all agent cost functions, though here formation of the cost matrix is assisted by the agent community: this was referred to as a hybrid scheme within Chapter 4.

11.2.3 Distributed Planning and Action: DPA

For decentralized planning, it is the task of each agent to evaluate their own cost function associated with moving to each possible array-point, and then use this cost function within the agent community to help formulate a decision. Within Chapter 4, auction processes were presented as a mechanism for reaching agreement between self interested agents. Applied to the interferometer scenario, each available array-point will be placed for auction and the agents only bid for their favored position: that for which the associated cost function is minimal. If no agent makes a bid then the next available array-point is placed for auction and the rejected position is placed for auction at a later stage when it may be more desirable.

Instigation of the auction is made by the agents themselves: any agent is able to request an auction, though there must be a consensus within the agent community for the auction to take place. Auction upon consensus, or rather "unanimous agreement", was implemented here rather than "collective agreement", since the triggers necessitating auction investigated are unambiguous. The triggers to hold an auction is based upon current percepts relating to event triggers and these will be discussed in more detail within Chapter 13.

An interesting problem relating to the concept of agent bidding is how to allocate funds in a fair manner such that agents can dominate auctions for which a winning bid is most desirable. The strategy implemented in this instance is to allocate funds according to the difference in the cost function between the two most desirable array-point allocations remaining in the auction: these two positions correspond to the two minimal costs developed within (11.2) or (11.3). Such a method fairly encodes the ability for an agent to dominate an auction if the penalty for a failed bid is severe. Upon auction of an array-point, each agent participates in a 'one-shot' bid auction[7] by placing their maximum bid if they are interested in the array-point being auctioned. Interest in an array-point is determined by searching for the position corresponding to the minimal cost within the generated cost function: an agent is interested in the position associated with the minimal cost. Note that merely being interested in, and bidding for, a specific array point does not guarantee attainment of this position; a competing agent may have more funds available.

It is assumed that within the auction, all agents are 'honest' regarding the formulation of their available funds, though during the auction, the manager may or may not compete in a fair manner. Fairness in this respect is with regards to altruistic behavior of the auction manger. Instances of less altruistic allocation, wherein the agent auction manager immediately takes their preferred array-point allocation, regardless of this being beneficial for the community or not, may also be implemented and this shall be referred to as a precedence auction.

There are four separate algorithms relating to auction protocol which must be followed by the agents within the community. These will now be detailed.

Main Auction Algorithm The main auction protocol relates to the complete auction framework and is that which each agent within the community must adhere to. The algorithm is as follows:

1. Instance of AAP allocation detected by agent. Such instances may be detection of a failed agent within the community or reaching a set-point within the

[7] Within a 'one-shot' auction, each agent has a single bid attempt for each array-point and the winning agent is that with the highest bid. The bidding process is non-incremental and thus reduces communication rounds.

mission.

2. Agents signal to the community that an auction is desired.
3. Upon community consensus, an auction is instigated.
4. Each agent computes their own cost function relating to their current state and the set of optimal array points, Λ_m. From this their available funds are determined: this is completed using the methods presented previously.
5. A single agent is selected to act as the auction manager.
6. It is at this point where the protocols for auction manager and auction participant diverge: the protocols for each of these will be detailed separately. There are three possibilities which enter at this juncture:
   (a) The agent is an auction manager operating in an altruistic manner.
   (b) The agent is an auction manager operating in a precedence manner.
   (c) The agent is an auction participant.

   The agents follow their respective protocols and these will be described subsequently.
7. Upon completion and finalization of the auction, the agents move to their allocated positions.

Altruistic Auction Manager Algorithm If selected to be an auction manager, operating in an altruistic mode, the agent must adhere to the following protocol:

1. From the set of optimal array points, Λ_m, an auction list of available AAPs is created and transmitted to each of the agents participating within the auction.
2. The auction manager awaits confirmation from each of the participating agents that the auction list has been received.
3. The uppermost location within the AAP auction list is placed for auction.
4. If interested, the auction manager places it's maximal bid for the current auction item.
5. The auction manager sequentially requests and receives maximal bids for the current auction item from agents within the community.
   (a) If no agents bid for the current AAP being auctioned, the particular location is not assigned. A signal is broadcast to the agent community indicating that the position was not taken and the declined AAP is placed at the bottom of the auction list. The auction manager then returns to point 3.

(b) Should one or more agents bid for the current AAP being auctioned, the winning agent is that which placed the highest bid. In the event that two or more agents place the same bid, the winning agent is that which made the first bid within the auction round[8].

(c) If relevant, the agents are informed of their success within the auction. Agents who did not participate in the auction are informed that the AAP has been taken.

(d) If successfully auctioned, the AAP is removed from the auction list.

(e) Whilst the auction list is not empty, the auction manager returns to item 3 and the process repeats until all AAPs have been allocated.

6. Upon complete allocation of all AAPs, the auction manager signals to the community that the auction process has been completed: this is the end of the auction manager role and it is at this point that all agents follow the same action.

Precedence Auction Manager Algorithm If selected to be an auction manager, operating in a precedence mode, the agent must adhere to the following protocol:

1. From the set of optimal array points, Λ_m, an auction list of available AAPs is created and transmitted to each of the agents participating within the auction.
2. The auction manager awaits confirmation from each of the participating agents that the auction list has been received.
3. The uppermost location within the AAP list is placed for auction.
4. The auction manager sequentially requests and receives maximal bids for the current auction item from agents within the community.

   (a) Upon the first iteration of the auction process, the auction manager extracts it's favored location from the AAP auction list and moves to point 4d.

   (b) If no agents bid for the current AAP being auctioned, the particular location is not assigned. A signal is broadcast to the agent community indicating that the position was not taken and the declined AAP is placed at the bottom of the auction list. The auction manager then returns to point 3.

[8] Whilst it is possible to encode additional information to permit favoritism within the auction process, such as the number of failed bids within the current auction round, this was not completed in this work.

(c) Should one or more agents bid for the current AAP being auctioned, the winning agent is that which placed the highest bid. In the event that two or more agents place the same bid, the winning agent is that which made the first bid within the auction round[9].

(d) If relevant, the agents are informed of their success within the auction. Agents who did not participate in the auction are informed that the AAP has been taken.

(e) If successfully auctioned (or taken by the auction manager), the AAP is removed from the auction list.

(f) Whilst the auction list is not empty, the auction manager returns to item 3 and the process repeats until all AAPs have been allocated.

5. Upon complete allocation of all AAPs, the auction manager signals to the community that the auction process has been completed: this is the end of the auction manager's role and it is at this point that all agents follow the same action.

Auction Participant Algorithm If the agent is operating as a participant within the auction process, the agent must adhere to the following protocol:

1. The agent awaits for the auction list to be broadcast by the auction manager: this will enable comparison of the point being auctioned with it's internal database and associated cost function and fund list.
2. The agent acknowledges receipt of the auction list from the auction manager.
3. The agent awaits an auction update relating to which AAP is being auctioned.
4. Upon receipt of the current auction item, the agent compares the item to it's current cost matrix to investigate if the AAP is of interest. The agent is interested if the item being auctioned corresponds to the minimal entry within it's cost matrix.
5. If the AAP being auctioned is not of interest:
   (a) The agent signals that it will not be participating within the auction round.
   (b) The agent awaits an auction update relating to the outcome of the auction which the agent declined to enter: if the point is not auctioned this must be noted; if the point is auctioned successfully, the agent must modify it's

[9] Whilst it is possible to encode additional information to permit favoritism within the auction process, such as the number of failed bids within the current auction round, this was not completed in this work.

cost matrix and associated funds accordingly, such that its knowledge of remaining AAPs is current.

(c) The agent returns to item 3.

6. If the AAP being auctioned is of interest:

   (a) The agent places it's maximal bid, as permitted by the funds allocated for the particular AAP, and awaits a response from the auction manager.

   (b) If the agent fails to win the bid, the unsuccessful agent must modify it's cost function and associated funds relating to removal of the auctioned AAP: the interests of the agent have now changed and the agent returns to item 3.

   (c) If the agent wins the bid, it no longer competes in any future auctions and exits to item 7.

7. The agent awaits an auction update relating to the completion of the auction process.

For clarity, a flow chart of agent action following the auction protocols is provided within Figure 11.1.

A consideration here regards the ability to refer to the above auction process as distributed when, by selecting a single agent to act as an auction manager, we are reducing the process to a formulation very similar to that described as a hybrid scheme. Indeed, had the action manager requested all agent cost functions immediately, the allocation problem could have been solved immediately using the previously discussed centralized method through formulation of a cost matrix[10]. However, in this instance we are using repeated agent communications and the concept of agent 'funds', rather than direct use of the individual cost functions, to formulate a global decision; the funds are resultant from manipulation of the cost functions generated. Due to these reasons, reference to the auction protocol as a distributed method, shall remain.

11.3 Mission Duration Maximization

Maximizing the mission duration requires an equal expenditure of agent resources, in this consideration ΔV, such that the maximum number of agents remain operational for the longest time possible. Since there are differing ΔV costs associated with differing array point allocations, and motion between these points results in a ΔV depended upon origin

[10] In no way would such an implementation divert from a MAS application, since it would provide an example of a hierarchical agent system.

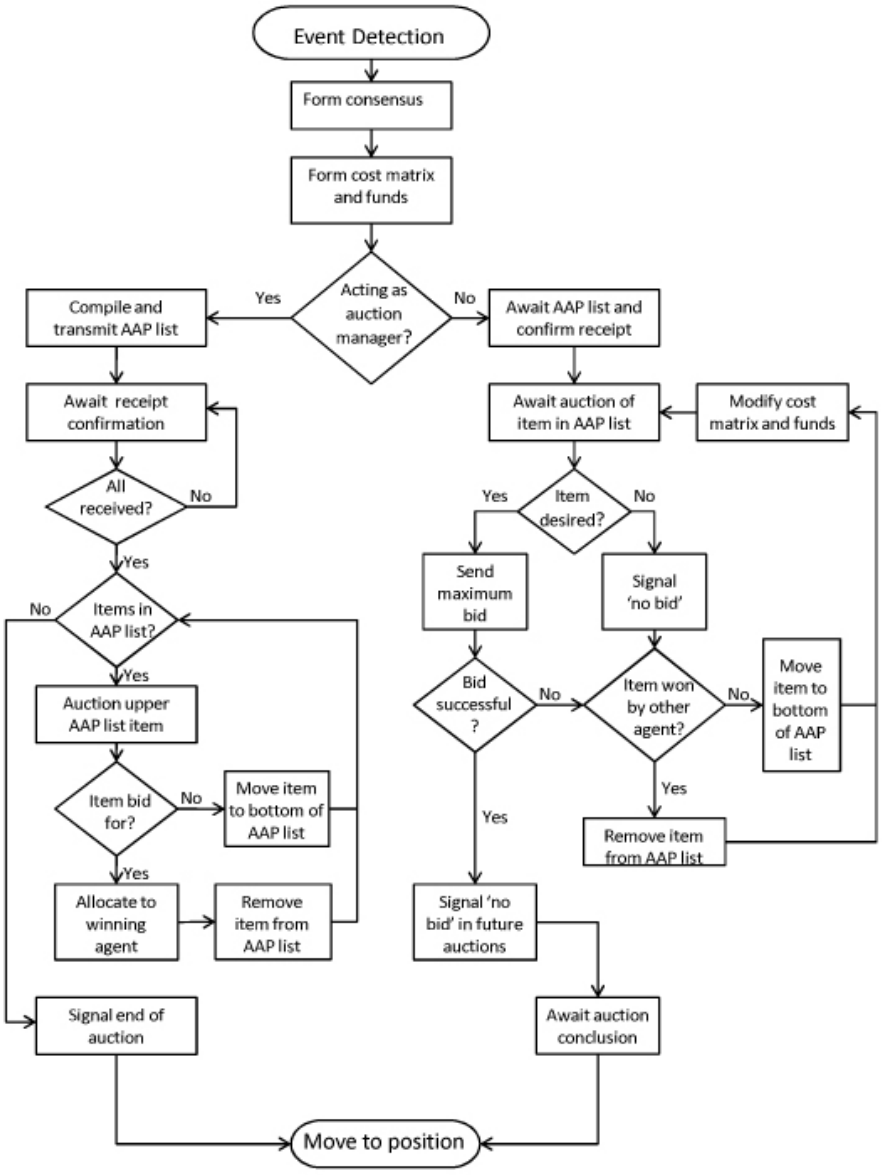


Fig. 11.1: Flow chart for agent auction decision method upon event detection. The methods and metrics for event detection and auction manager nomination are not detailed.

and destination, it is desired to determine the sequence of AAP allocations to maximize ΔV resources across the agent system.

To reduce the complexity of the minimization problem, it will be assumed that AAP relocation will only occur on each instance of halo orbit completion; that is, upon completion of an orbit cycle, a new stage of the centralized scheme will be entered or a new auction will be instigated.

11.3.1 Centralized Planning With Distributed Action: CPDA

The global optimization of the relocation problem, where we are now interested in not only the instantaneous cost of allocation, but future costs involved with subsequent AAP alteration, may be constructed into a binary integer problem readily solvable through commercial solvers. In the formulation of this problem, some new parameters will be introduced:

Maintenance Cost The maintenance cost parameter is a $2D$ matrix, i in agents and j in array-points, encapsulating the $\Delta V^{maintain}(i,j,t)$ costs associated with agent i to maintain array-point j for a complete orbit and these orbits are indexed by time t. This cost information is initially obtainable through predictive integration methods applied to state equation (7.16) with $\delta\mathbf{x}_j$ held constant over an orbit period, and actual ΔV costs experienced by the agents thereafter.

Motion Cost The motion cost parameter is a $3D$ matrix: i in agents, j in array-points and k in array-points. This parameter encapsulates the $\Delta V^{move}(i,j,k,t)$ costs associated with agent i moving from array-point j to array-point k at time t. In a similar manner to the construction of the maintenance cost parameter, the initial values of the component entries may be formulated based upon predictive methods as presented in prior Sections, and replaced with experienced costs thereafter.

Defining a variable to denote the ΔV reserve for agent i at time t as $\Delta V^r(i,t)$, the objective is to maximize this variable across all agents subject to certain constraints, which can be mathematically prescribed as

$$max \sum_i \Delta V^r(i,t=n) \tag{11.5}$$

subject to:

$$\begin{aligned} \Delta V^r(i,t+1) &= \Delta V^r(i,t) - \Delta V1(i,t) - \Delta V2(i,t) \\ \Delta V^r(i,t=0) &= \Delta V^r(i,0) \end{aligned}$$

Maximization of this is achieved through prescription and modification of a 4 dimensional binary variable: i in agents, j in positions, k in positions and t in time; which we shall refer to as $AgentSwap(i,j,k,t)$. Within this variable, across all $AgentSwap(i,j,k,t)$, the entry is unary if at time t, agent i moves from position j to position k and zero if not. This variable is subject to the following constraints:

$$AgentSwap(i,j,k,t=0) = AgentSwap(i,j,k,0) \quad (11.6)$$

$$\forall(i,t)\sum_{j,k} AgentSwap(i,j,k,t) = 1 \quad (11.7)$$

$$\forall(i,j,t)\sum_{k} AgentSwap(i,j,k,t+1) = \sum_{k} AgentSwap(i,k,j,t) \quad (11.8)$$

$$\forall(j,t)\sum_{i,k} AgentSwap(i,j,k,t) \leq 1 \quad (11.9)$$

Item (11.6) enforces a particular starting point, item (11.7) restrains an agent to be in a location, item (11.8) enforces the fact that an agent can only move from where it was and item (11.9) enforces the condition that only one agent is to be stationed at an array-point at any time instant.

Consequently we may define agent and time specific ΔV costs as:

$$\forall(i,t)\Delta V1(i,t) = \sum_{j}\Delta V^{maintain}(i,j,t)\sum_{k} AgentSwap(i,j,k,t) \quad (11.10)$$

$$\forall(i,t)\Delta V2(i,t) = \sum_{j,k}\Delta V^{move}(i,j,k) AgentSwap(i,j,k,t) \quad (11.11)$$

Since we are seeking the generation of a variable indicating instances of AAP rearrangement indexed with time, an additional variable shall be introduced to enable a more natural observation of the optimal resource redistribution: indicating the position, j, of agent i at time t. This variable shall be denoted $AgentPos(i,j,t)$ and is dependent upon $AgentSwap(i,j,k,t)$, since $AgentPos(i,j,t)$ is defined as:

$$AgentPos(i,j,t) \equiv \sum_{k} AgentSwap(i,j,k,t) \quad (11.12)$$

Once the optimal relocation order has been determined through the centralized method, this information is stored and upon reaching a time increment within (11.12), the agents automatically redistribute to their respective AAP allocations for the next orbit.

11.3.2 Distributed Planning and Action: DPA

Application of a distributed method to the problem of agent system longevity, results in a similar implementation to that of agent redistribution resulting from element failure, presented previously. Here (11.3) is used repeatedly upon instances of orbit completion, wherein the agents hold an auction to decide if any agents should move into different positions during the forthcoming orbit.

It should be noted that the auction methods use cost functions to formulate a decision whereas the central method is concerned with ΔV costs associated with each possible AAP allocation and keeps track of the agent ΔV reserves throughout the target mission duration.

11.4 Execution of Decisions

In as much as the formulation of a decision varies between the centralized and distributed methods, execution of the resultant decision also varies.

Central Decisions All decision instances are themselves determined by a central agent. Decisions may be generated through either cost functions or direct ΔV costs, which may or may not be based upon information gathered from the agent community. Irrespective of the manner information is harvested, the decision is made based upon global knowledge and is constructed in the form of a plan, which may relate to a single instance or a sequence of instances. The central agent prescribes the plan directly to the concerned agents, which then complete their tasks. No other agents may influence the resultant decision and must act in accordance to the central decision once prescribed.

Distributed Decisions All decision instances are determined through a community consensus and upon agreement of a decision instance arising, the relevant auction protocol is entered based upon 'honest' bids and local knowledge only. Once the auction is completed, the agreed action between all agents is carried out: whilst similar to the centralized decision method in that no agent has the option to appeal, in each auction an agent is given a fair chance to secure their favored outcome.

11.5 Chapter Summary

This Chapter has presented agent skills for centralized and distributed decision methods, required for the prescription of agent resources and based upon methods applied to cost

functions generated for varying scenarios. The centralized methods result in a globally optimal solution by a single central agent, based upon knowledge of the complete problem which may or may not be self-generated. Distributed decision methods use auction techniques, such that only local knowledge is used by the agent community to form a fair decision relating to the current problem.

The following Chapter will investigate the agent skill sets developed within the last two Chapters, which relate to agent control, as presented within Chapter 10 and agent decision, as presented within this Chapter.

12. SPACECRAFT AGENT SKILL ANALYSIS

Evaluation of the spacecraft agent skills relating to state determination has already been addressed: this Chapter will evaluate the skill sets relating to spacecraft agent control and decision. As with the simulation results presented for the issue of state determination, within Chapter 9, the simulation results to be presented here serve only as an illustration of the theoretical results derived within Chapters 10 and 11, and cannot be considered absolute verification of the derived controller or decision method.

For the purpose of modeling controlled spacecraft agent motion within the CRTBP, a simulation environment was constructed within MATLAB (R2006b). The environment was based upon the dynamics located in the vicinity of the Sun-Earth/Moon $L1$ point, as were presented and tested within Chapter 6. The intention was to simulate a sampled continuous time environment, achieved by completing all dynamic modeling using continuous time methods[1], but dividing the continuous scheme into linked Sections of time duration h. Spacecraft agent percepts were taken at the end of each sampling period and used to obtain Kalman filtered state estimates, as presented within Chapter 9. Control action was determined through these state estimates, using the methods described in Chapter 10; itself based upon the current decision, as formulated by the methods presented within Chapter 11. The control output between sampling instants remains constant. This information flow is depicted within Figure 12.1.

The mother-satellite, from which the agents disperse, is assumed to be controlled to a reference orbit: the reference orbit is generated through a third order analytic Richardson approximation as presented within Chapter 6. Although a differentially corrected halo orbit could have been used for the mother-satellite trajectory, since the objective is robust control of the agents relative to the mother-satellite, it was not necessary for this implementation.

Agent ejection from the mother-satellite is assumed to be initiated through an impulsive release mechanism, such as a spring, imparting a force to each agent at the same time instant. As a consequence of non ideal release, this impulsive force gives rise to each

[1] Continuous time simulation was completed using the in-built MATLAB *ODE*113 solver, with relative and absolute tolerances for the solver specified to 10^{-13} and 10^{-22} respectively.

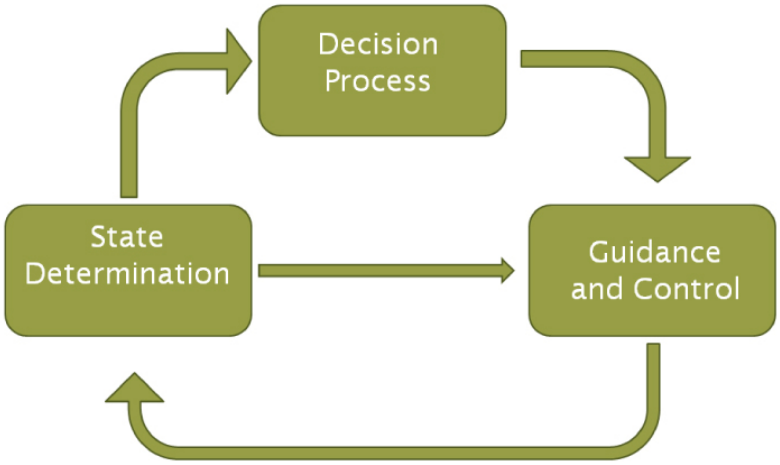


Fig. 12.1: The agent perception/action cycle in which decisions are integral to outputting the appropriate control action.

agent gaining a bounded translational velocity subject to a random bounded error and an additional random, bounded angular velocity.

12.1 Control Action

This Section will evaluate the performance of the control action, as presented within Chapter 10, and used by the agent community to obtain the desired interferometry formation having been ejected from the mother-satellite.

In addition to positional acquisition and maintenance of the required optimal array configuration, an attitude change corresponding to Euler rotations of $[45,10,0]^\top$ degrees about the x, y and z axes respectively was also introduced. Note that although notation using Euler rotations has just been used, this was primarily for reader benefit: this Euler rotation was transformed into a desired quaternion and the quaternion notation was maintained throughout the simulation.

For control, a sampling period of 0.2 seconds was used since this is a realistic time step for active vision situations. Each agent was considered identical with a mass of 10kg and inertial matrix of $\mathbf{J} = diag[2,1,0.7]$ kgm^2, chosen as representative values for a nano-satellite. Actuator outputs were restricted by force and torque constraints of 0.1N and 0.005Nm respectively[2]. Distance measurements, subject to gaussian white noise, were in-

[2] The actuator force and torque inputs were simulated as ideal, with no internal dynamics or delays:

troduced to yield an error bound of 0.5cm. The release mechanism was assumed to impart a translational velocity to each agent of magnitude $0.2m/s$, with a random error bounded between $\pm 0.05m/s$ and a random angular velocity bounded between $\pm 0.02rad/s$. For the reaching law constraint, (10.12), values of 0.01 and 0.7 were used for the the constants ε and ς respectively with a c_v value of 0.3 being implemented within (10.3).

Simulation output for mission initiation, which is inclusive of agent ejection, AAP acquisition and early stages of AAP maintenance, is displayed within Figures 12.2 to 12.6. Figure 12.2 shows the complete motion of the agent community upon ejection from the mother-satellite: the path of the mother-satellite is shown in red and the paths of the individual agents in black, with the final position at the end of the simulation denoted by circles. Figure 12.3 indicates the maximum and minimum agent proximities during the dispersal phase. Figure 12.4 looks at the resultant agent community configuration in the context of an interferometer array, showing the $2D$ aperture locations and the u,v-plane coverage, generated at the end of the AAP acquisition phase. Figure 12.5 shows the sliding surface motion, for both σ_i^1 and σ_i^2: for sake of clarity the norm of the respective sliding surfaces is given. Figure 12.7 shows the controller output during the agent distribution phase for a single agent: the upper plot is that of force command and the lower plot torque demand. As with Figure 12.5, it is the norm of both the force and torque demands that is shown. Figure 12.6 shows the quaternion progression for a single agent upon ejection from the mother-satellite: the upper plot depicts the time evolution of the normalized quaternion vector component and the lower plot that of the quaternion scalar component.

Figures 12.2, 12.3 and 12.4 indicate that the agent system skills permit acquisition and subsequent maintenance of the desired global formation, without inter-agent collision. Upon non-ideal release, the optimal Golay-10 array is formed, in which the minimal agent proximity does not reduce to below that of the stowed configuration (0.3 meters). One should note that the agent proximity relates only to the distance between the center of mass of the respective agents: physical dimensions and attitude transformations are not taken into account.

Figures 12.5 to 12.7 relate to a single agent only but are representative of each agent within the community. Figure 12.5 indicates the sliding surface motion, which exhibits the desired exponential reaching rate, as specified during controller development: it is evident that the sliding surfaces progress to zero within finite time and are subsequently maintained. Figure 12.6 indicates the quaternion progression, which is an element of the second sliding surface σ_i^2. One can see that the vector component of the quaternion progresses to zero and that scalar component to unity: which is zero attitude error by

the requested actuator input was applied both instantly and at the desired magnitude. Saturation was only applied to the upper limit of thruster output.

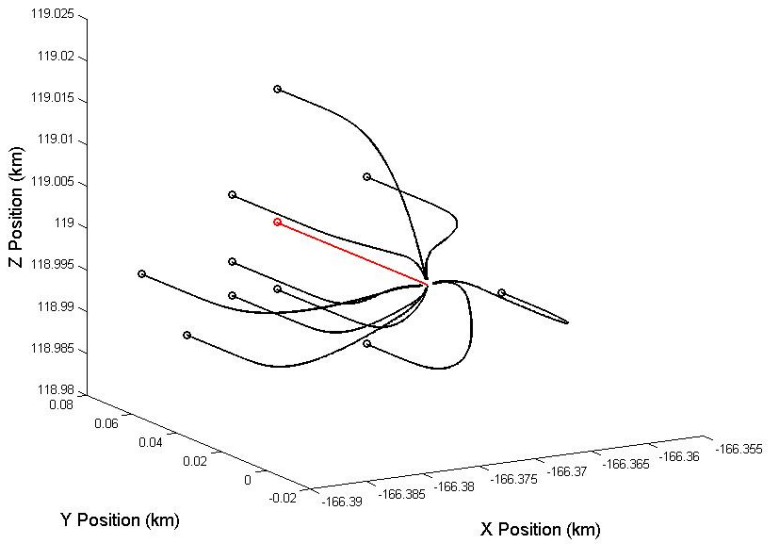


Fig. 12.2: Complete agent motion dispersing from centralized mother-satellite into a Golay-10 formation, h=0.2s. The red trace indicates the path of the mother-satellite and the black traces represent the individual agent paths: the circles represent final positions at the end of simulation run time.

definition. Actuator usage for the agent is shown within Figure 12.7, which indicates that upon initial dispersal and a relatively high degree of control action, the command signal drops to a steady state value required only to combat the relative accelerations resultant from maintenance of a non-ideal orbit. The maintained agent-system, for a significant portion of the halo orbit, is shown within Figure 12.8.

The time discretization period of $h = 0.2$ seconds was chosen as being representative of realistic sampling times: reduction of the sampling frequency was observed to cause a reduction in controller performance, leading to slower reaching times and eventually discretization chatter. Sampling periods between 0.5 and 0.8 seconds allowed for the attainment of a steady state control response upon formation of the array; sampling periods greater that this were observed to result in discretization chatter. Although the chatter was bounded within a domain of the prescribed sliding surface and hence, relating to the stringent sliding surface requirements given within [155], we are satisfying all requirements for sliding motion in discrete time; such chatter relates to inefficient use of control resources and consequently reduced mission lifetime.

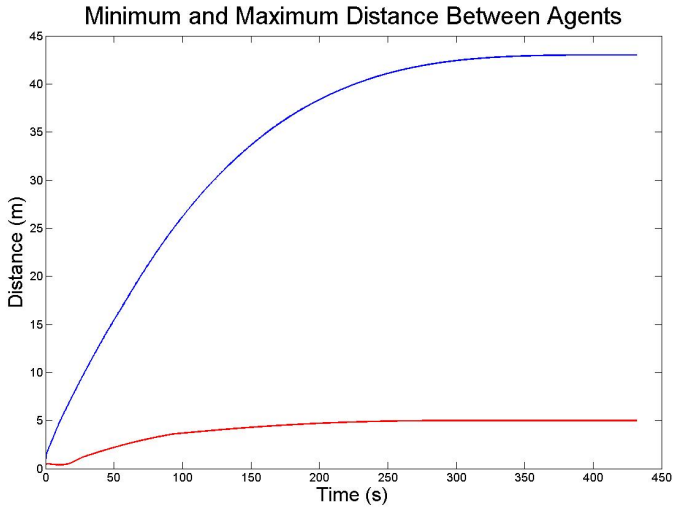


Fig. 12.3: Time evolution of agent proximity upon dispersal considering minimum and maximum proximities only: red line corresponds to that of the minimal distance between agents and the blue line to the maximum distance.

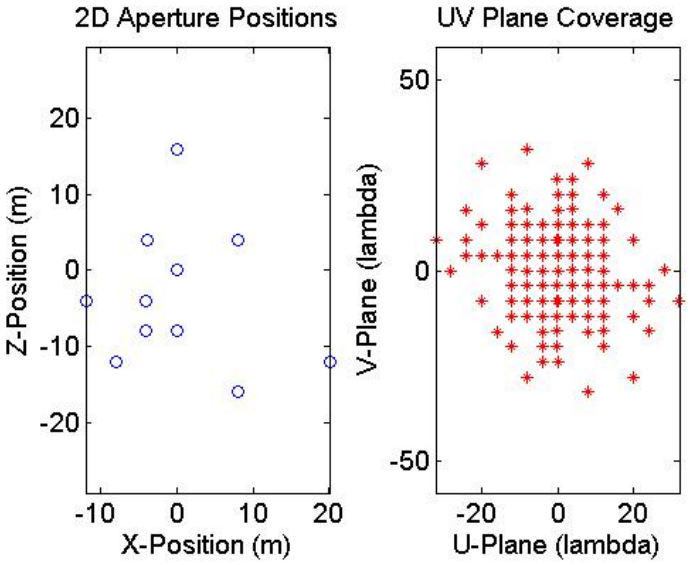


Fig. 12.4: Final 2D aperture locations and resultant u,v-plane coverage for the agent community, depicted in the left and right plots respectively: a Golay-10 formation is achieved.

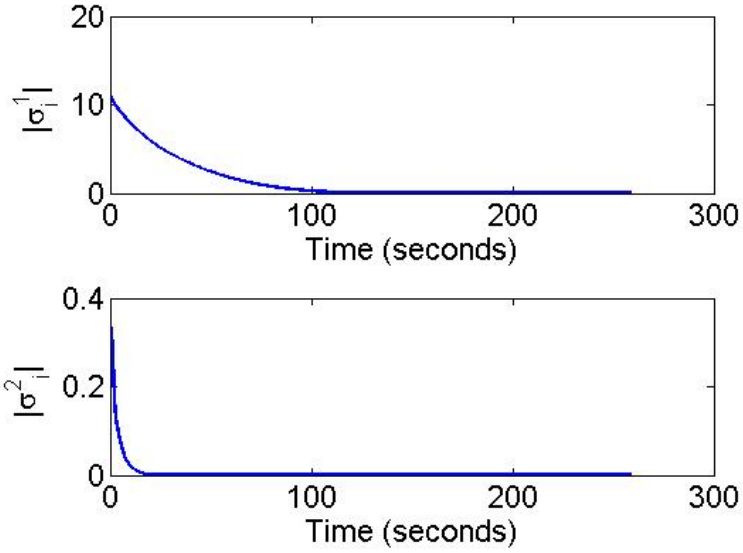


Fig. 12.5: Norm of sliding surfaces, σ_i^1 (translational) and σ_i^2 (rotational), for a single agent during the initial dispersal phase and subsequent maintenance, h=0.2s.

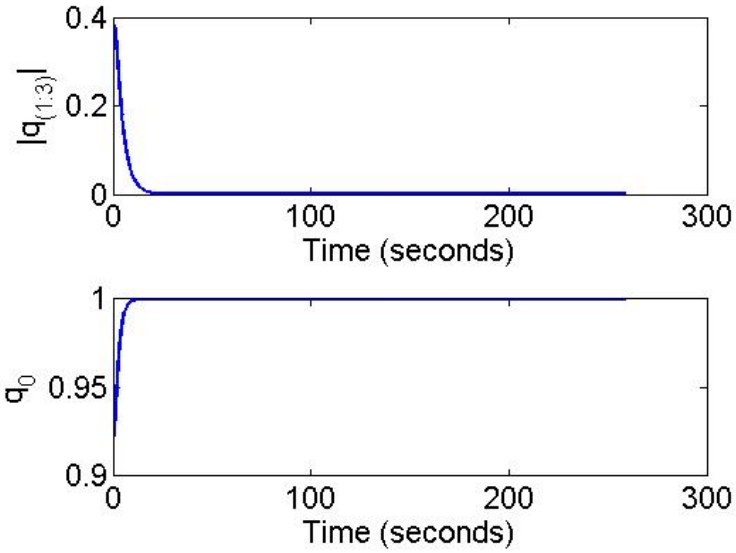


Fig. 12.6: Quaternion response during initial attitude regulation: the upper plot represents the norm of the quaternion vector component and the lower plot that of the scalar component.

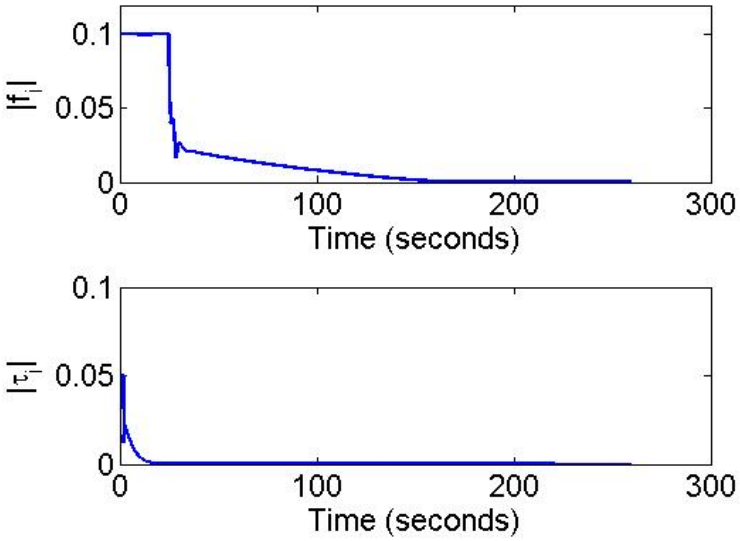


Fig. 12.7: Agent controller output during initial distribution and position acquisition/regulation phase: the upper plot is that of force and the lower plot is that for torque. Force and torque limitations of 0.1N and 0.005Nm were applied to each agent of mass 10kg.

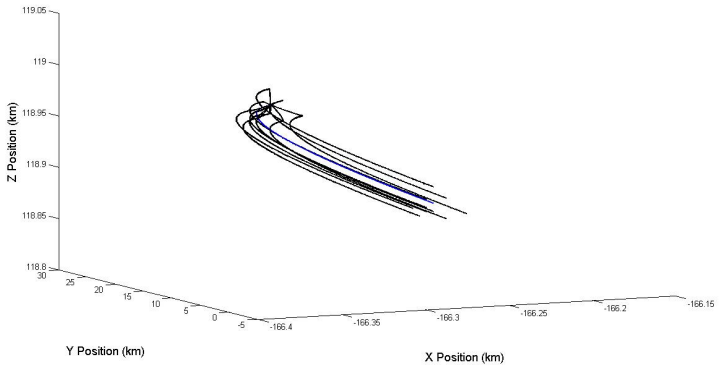


Fig. 12.8: Dispersed swarm motion for a significant fraction of the halo orbit

12.2 Decision Methods

This Section will present the decisions, and the consequences of these decisions, made by the multi-agent system regarding AAP allocation and the desire to maximize the mission life time during simulation. Three decision instances will be investigated: initial AAP allocation upon agent system dispersal, AAP reallocation in the instance of agent failure and AAP reallocation in order to maximize the mission. It is assumed here that centralized methods, based upon complete knowledge of the minimization criteria, are capable of forming *ideal* decisions relating to single instances of assignment and as such we are seeking the distributed methods to form identical, or similar, decisions. This is not necessarily true for instances of multiple assignment, as consequent from mission maximization: here we assume that the central method is capable of forming a decision which satisfies the requirements, though the solution may not be ideal in some respects. It is here where a comparison with distributed solutions will be based upon alternative figures of merit, such as agent community health and decision times.

Agent state data was extracted directly from the MATLAB simulation and used to form the pertinent cost functions that were developed within Chapter 11. In all auction instances, the auction manager was selected randomly from the set of functional agents within the agent community.

12.2.1 AAP Allocation

The initialization of the agent system configuration commences with ejection from the mother-satellite and deciding upon the initial AAP allocation, for which both centralized and distributed decision methods have been presented relating to the cost function given within (11.2). The Hungarian and auction based decision processes were applied to the agent ejection scenario, for 100 instances, wherein each agent was initiated with a fuel fraction of unity and a state resultant from non-ideal ejection from the mother-satellite.

The results from these simulations are displayed in Table 12.2.1, where each row represents a decision method: the upper row relates to the centralized (optimal) allocation; the central row to an altruistic agent auction, wherein each position is placed for auction; and the bottom row to a precedence auction, wherein the agent holding the auction automatically takes their favored array-point allocation prior to the remaining locations being auctioned. The average and maximum % optimal allocation indicates the percentage of identical placements made by the decision process under consideration, when compared to that of the optimal method. Within each row, this parity with the optimal method is displayed with the average assignment cost and maximum possible cost that could have

been incurred, which relates to the least optimal allocation of AAP allocations[3].

Method	Average %Optimal Allocation	Maximum %Optimal Allocation	Average Cost of Assignment	Maximum Possible Cost
Centralized	100	N/A	8.069	11.409
Distributed -altruistic	66.8	80	8.237	11.409
Distributed -precedence	36.8	60	8.421	11.409

Tab. 12.1: Table of decision results for AAP allocation, based upon initial dispersal from the mother-satellite, for 100 repetitions.

Whilst the quantitative data shown within Table 12.2.1 permits evaluation of items such as the percentage of optimal allocations, it does not allow for visualization of the decision processes and what response these decisions trigger. Figure 12.9 displays the output system motion as a result of both centralized and distributed decision processes: the left image is the output system motion based upon a centralized and optimal decision; the right image is the system motion consequent from a distributed decision process, which in this instance was from a 50% optimal allocation match and with an optimal:distributed allocation cost ratio of 10.158:10.391.

12.2.2 AAP Reallocation

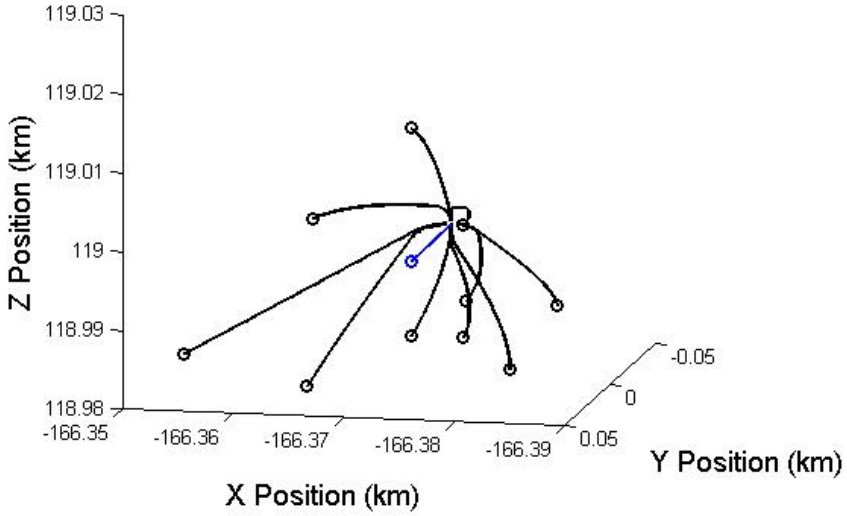
In the presence of total agent loss within the agent system, commodity redistribution is required to maintain optimality of the interferometer array. As discussed previously, this is a similar decision process to that of system initialization, wherein allocation of new AAPs is completed based upon the cost function given within (11.3), relating to the optimal set of reduced array points. In an identical manner to the evaluation of initial AAP allocation, a scenario was developed to assess the effectiveness of an auction system in comparison to a centralized method. Within the investigated scenario, subsequent to agent ejection, the initial AAP allocation was made by a centralized method. Upon allocation assignment, the agents take their positions and maintain them for a complete orbit. The propulsive requirements for both of these processes are accounted for within the respective agents' remaining fuel fractions and we are now in the position where agents have differing fuel

[3] The maximum possible total cost for each AAP allocation process was calculated using a centralized process, wherein the objective was to maximize the total incurred cost through selective allocation.

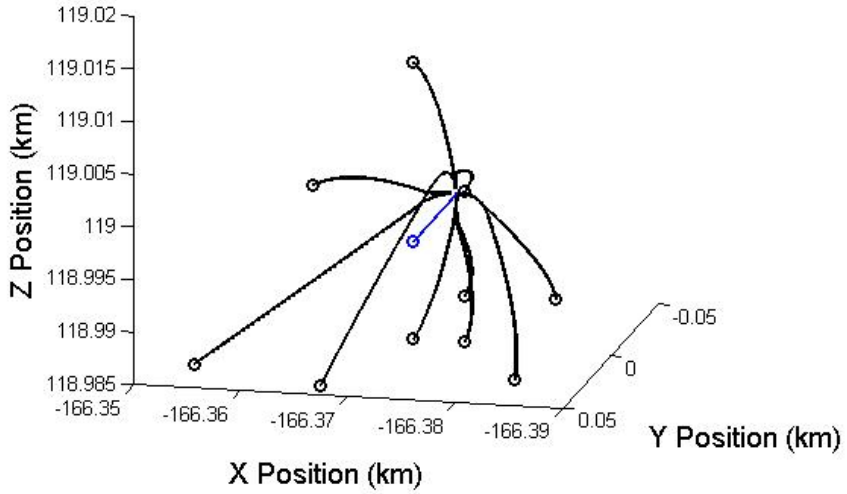
reserves. At this point, a random agent is assumed to fail, at which time a redistribution is required based upon the remaining functional agents' current positions, motion to viable array points and maintenance of the array point for another complete halo orbit. This process was repeated for 100 instances of random agent failures and a new set of AAP allocations determined by both centralized and distributed mechanisms; the results of these simulations are displayed in Table 12.2.2, with an exemplar graphic comparing the resulting redistribution decisions shown within Figure 12.10.

Method	**Average %Optimal Allocation**	**Maximum % Optimal Allocation**	**Average Cost of Assignment**	**Maximum Possible Cost**
Centralized	100	N/A	116.759	127.716
Auction -altruistic	66.22	100(7)	117.662	127.716
Auction -precedence	48.67	77.78	120.032	127.716

Tab. 12.2: Table of decision results for AAP reallocation, for 100 repetitions, based upon random agent failure after completion of a halo orbit. The altruistic auction achieves the maximum possible optimal allocation: the number of instances of this optimal allocation are enclosed within parentheses.

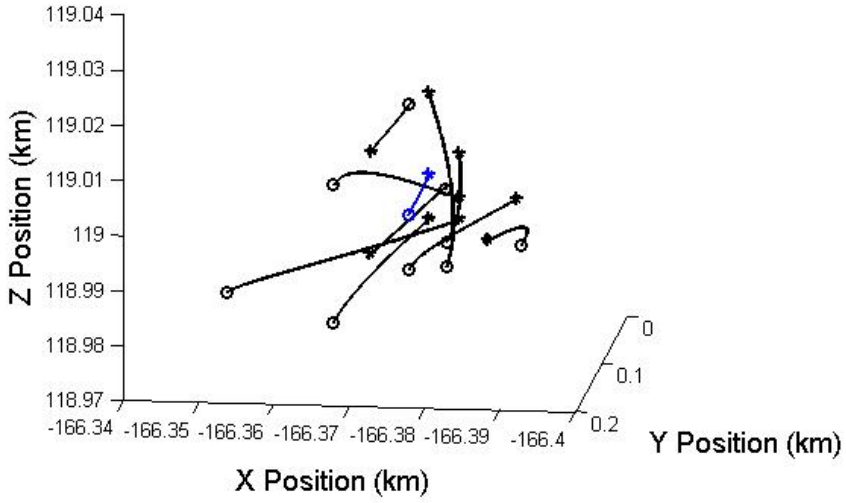


(a) Centralized Decision

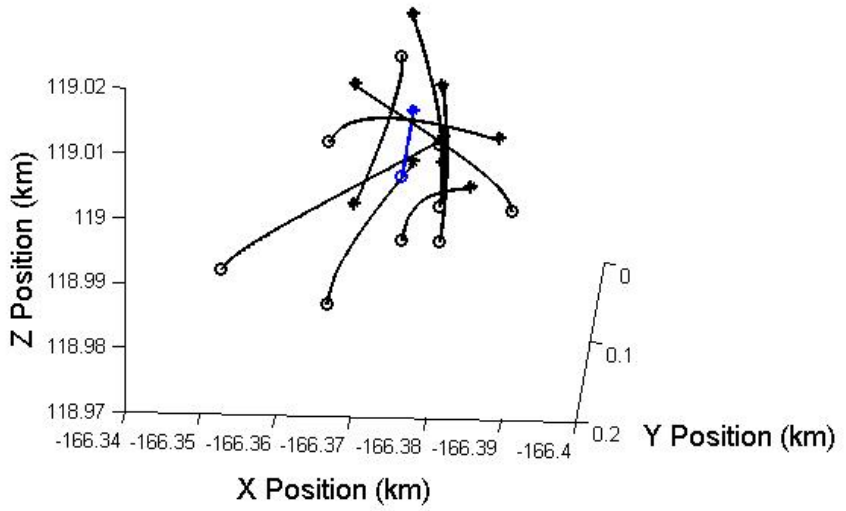


(b) Distributed Decision

Fig. 12.9: Visualization comparison of system output by use of central (upper) and distributed (lower) decision process for initial dispersal from the mother-satellite. The path taken by the respective agents is indicated by the individual line traces, terminating at the circular point.



(a) Centralized Decision



(b) Distributed Decision

Fig. 12.10: Visualization comparison of system output by use of central (upper) and distributed (lower) decision process for redistribution upon agent failure. Agent redistribution commences at points denoted by an asterisk and completes at points denoted with a circle; the path taken by each agent is shown by line traces connecting start and end points.

12.2.3 Mission Maximization

Mission maximization was considered for two scenarios involving formation of the Golay-10 array configuration, based upon differing scales applied to the coordinate points: a large aperture interferometer, with the optimal array point locations scaled to kilometers and a small aperture interferometer, with the optimal array point locations scaled to meters. This was done to enable investigation of both techniques to short and long term problems.

To generate a centralized decision, the optimization problem given within Section 11.3.1 was written using AMPL: this language was selected since AMPL allows prescription of a minimization problem in a natural format through model and data files; AMPL can then format the prescribed problem to be passed into a variety of commercial solvers [161, 162]. Such a methodology allows focus to be on the high level problem, rather than complexities of data structuring required for a particular solver. In actuality, the problem formulation given in Section 11.3.1 is precisely that implementable by AMPL, as can be seen when comparing (11.5) through (11.12), to the AMPL *.mod* file given within Appendix C. The associated AMPL data file was constructed using data extracted from the MATLAB simulation. Once solved, the generated sequence of AAP allocations represents the optimal solution for mission maximization.

A point of note is that the student editions of AMPL and CPLEX are limited to very small problems involving no more than 300 variables or constraints; the problems to be investigated here are significantly larger than this limit[4]. It is possible to submit problems online to the NEOS server in an AMPL format, with the solution being provided by email, however these problems are restricted in process time [166]. Tomlab[5] provides a comprehensive suite of optimization products, which include AMPL and a CPLEX solver accessible through MATLAB: it was this tool suite which was used in all optimization runs. Interface with CPLEX was preferred because trial experimentations proved CPLEX to be the most suitable solver, through its capability of dealing with the binary allocation variables most effectively.

Formulation of the distributed decision process relating to mission maximization followed an identical process to that of all previous auction techniques, based upon the cost function given within (11.3) being solved though the presented auction methods. Position auctions were initiated upon each complete orbit, with an auction manager being selected

[4] Optimizing agent position allocations over 150 orbit instances involves 168010 variables and 35920 constraints.

[5] Tomlab is a powerful optimization platform and modeling language for solving applied optimization problems in Matlab [167]. It is capable solving problems written using AMPL and can interface a variety of industry standard solvers.

randomly from the set of functional agents: both altruistic and precedence auctions were completed, in addition to position allocation based upon the Hungarian method, for comparison to the results produced by the centralized optimal method. It should be noted that application of the Hungarian method to this problem does not represent an optimal scheme since it is applied iteratively to the current state of the system and does not consider all possible agent position allocations throughout the mission to achieve an optimal end result; one could argue however that the application of a Hungarian method does represent the outcome of an ideal auction process.

Since it would be ineffectual to analyze correlations between position allocations upon each decision instance, the variable to be evaluated here is the fuel level (more specifically the ΔV remaining) of each agent. Of interest in particular, is both the ΔV variation during the mission life and the standard deviation of all agent ΔV reserves; an associated point of interest is the time taken for the decisions to be formed. A key point to note is that the centralized method directly uses ΔV costs to plan an AAP positioning sequence, whereas the auction methods use a cost function to form a decision applicable to a single instance.

Table 12.2.2 contains ΔV statistics resultant from centralized and distributed methods applied to the task of mission maximization for two problems: a small scale problem, representative of formation and maintenance of a large aperture interferometer and a larger scale problem representative of a small aperture interferometer. Note that scale here refers to the size and complexity of the problem, not the size of the aperture being formed. Presented within the table is the maximum standard deviation between agent ΔV levels, the number of agent losses[6] and the time taken for the respective method to solve the problem[7].

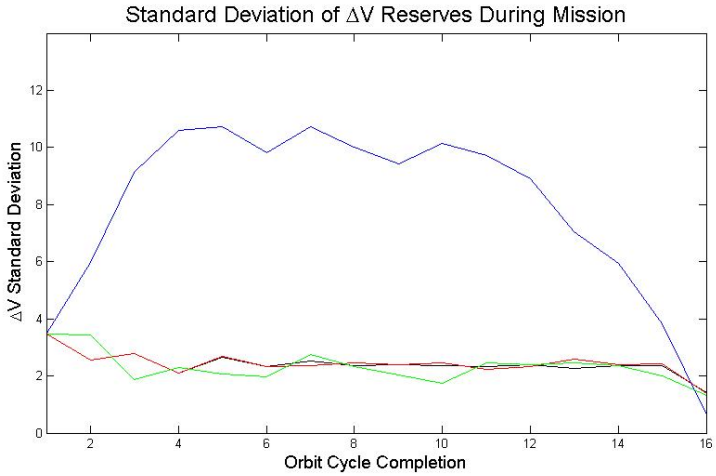
Figures 12.11(a) and 12.11(b) are graphs of agent ΔV standard deviations, resultant from centralized and distributed decision methods applied to the small and large scale problems respectively. The blue trace represents agent ΔV standard deviations as a result from the centralized decision method, the red, green and black traces represent the altruistic, precedence and Hungarian based auction mechanisms respectively.

[6] An agent is assumed to be lost when it's ΔV reserve reaches zero.

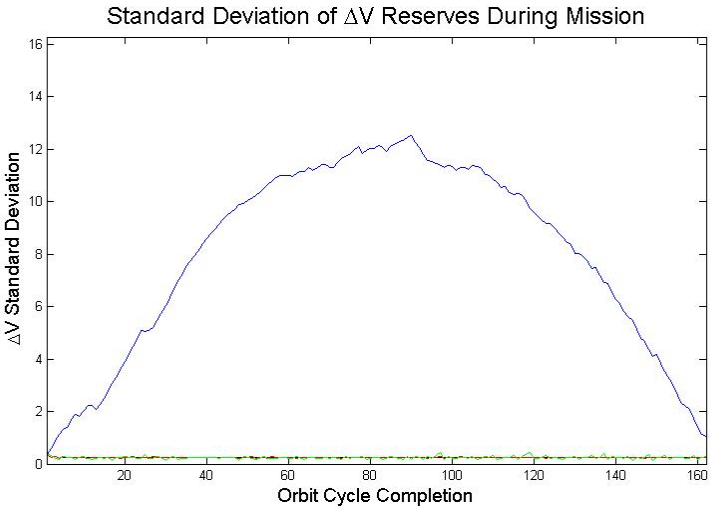
[7] Centralized optimization processes were computed on a Windows platform using an Intel Core 2 Duo 2.66GHz processor and 4GB of RAM

Method	**Problem Size (orbits)**	**Maximum Standard Deviation**	**Agent Losses**	**Solution Time (minutes)**
Central	15	11.33	0	6.4
	162	13.21	0	2332.7
Auction (altruistic)	15	3.67	1	< 0.1
	162	0.37	1	< 0.1
Auction (Precedence)	15	3.67	2	< 0.1
	162	0.41	1	< 0.1
Auction (Hungarian)	15	3.67	2	< 0.1
	162	0.37	1	< 0.1

Tab. 12.3: Table of ΔV statistics resultant from centralized and distributed methods relating to mission maximization.

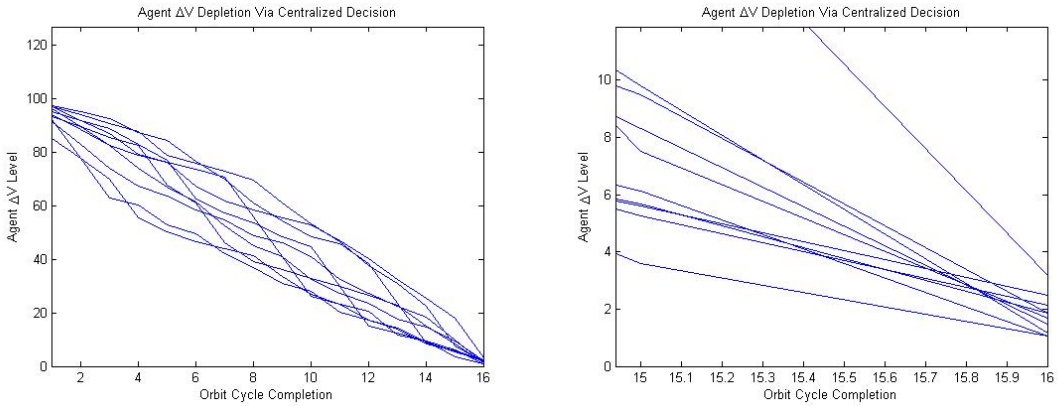


(a) Small Scale Problem: 15 orbits



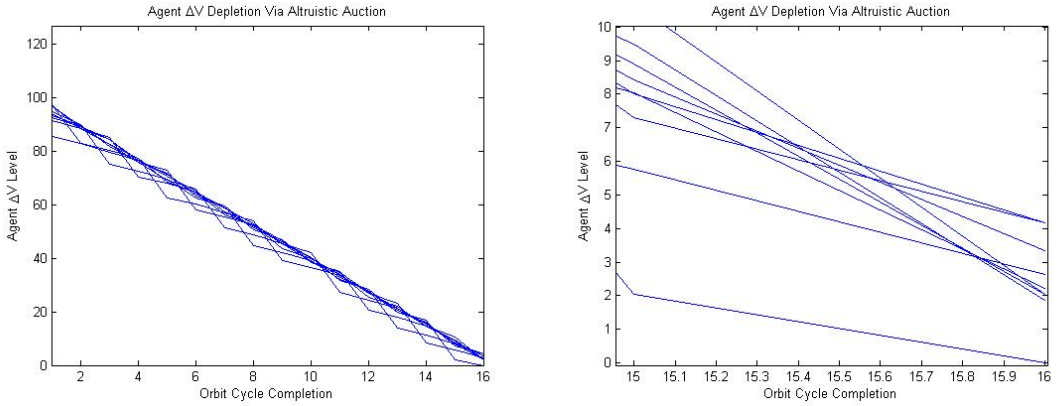
(b) Large Scale Problem: 162 orbits

Fig. 12.11: Graph of agent ΔV standard deviations over for small and large scale problems: the blue trace represents standard deviations resultant from a centralized decision process; the red, green and black traces represent auction methods applied to the same scenario.



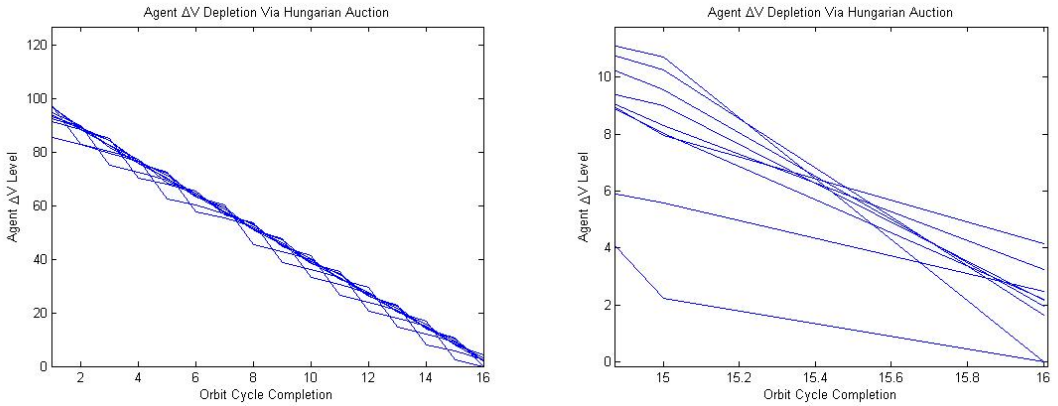
(a) Agent ΔV flow over whole mission (b) Zoomed view of end agent ΔV reserves

Fig. 12.12: Graph of agent ΔV flow based on utilization of centralized decision processes for a small scale problem.



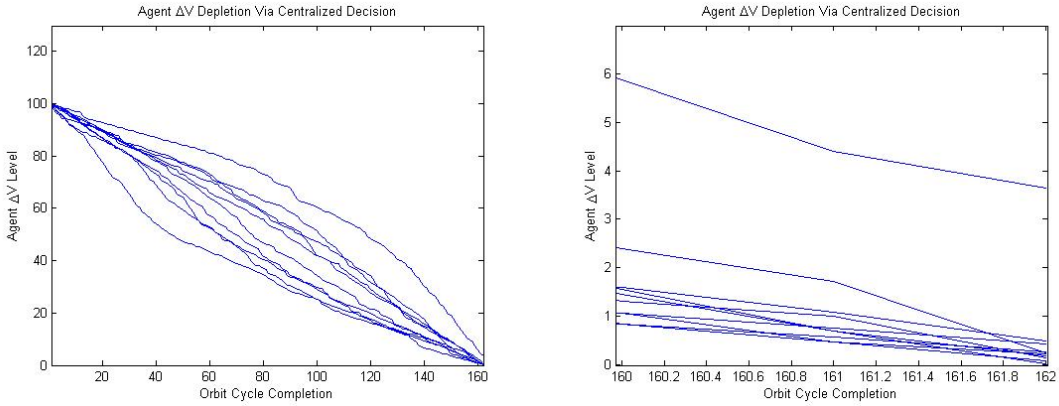
(a) Agent ΔV flow over whole mission (b) Zoomed view of end agent ΔV reserves

Fig. 12.13: Graph of agent ΔV flow based on utilization of an altruistic auction processes for a small scale problem.



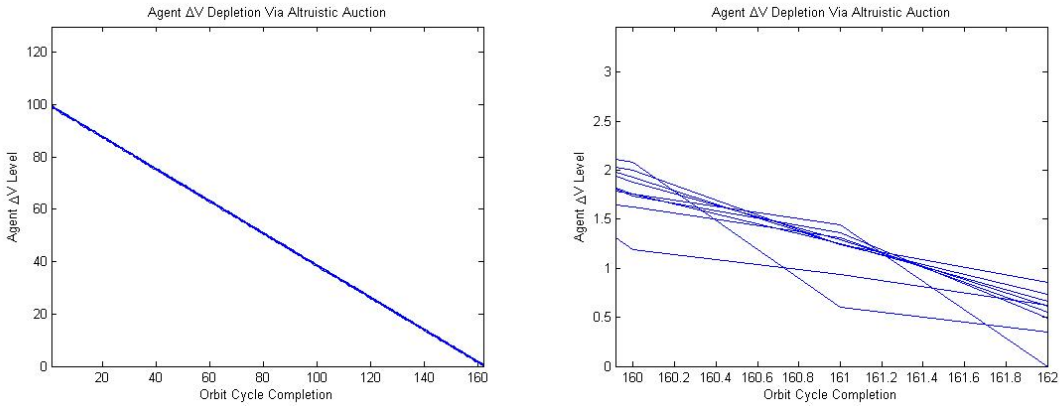
(a) Agent ΔV flow over whole mission (b) Zoomed view of end agent ΔV reserves

Fig. 12.14: Graph of agent ΔV flow based on utilization of a Hungarian auction processes during a small scale problem.



(a) Agent ΔV flow over whole mission (b) Zoomed view of end agent ΔV reserves

Fig. 12.15: Graph of agent ΔV flow based on utilization of centralized decision processes for a large scale problem.



(a) Agent ΔV flow over whole mission (b) Zoomed view of end agent ΔV reserves

Fig. 12.16: Graph of agent ΔV flow based on utilization of an altruistic auction processes for a large scale problem.

12.3 Evaluation

Having presented the decisions and ΔV consequences for the agent system, the effectiveness of centralized and distributed decision making process will now be discussed. In relation to analysis of auction methods for agent decisions, we are comparing the results achieved through auction methods to those formed by a central process which is assumed to make an ideal decision. For auctions relating to initial agent deployment and redeployment, 100 repetitions with differing pre-conditions were completed, with the intention of forming a comprehensive data set with which one could compare decision methods fairly. For the auctions relating to mission maximization, auction processes were applied to the long term mission and initiated upon each passage of original halo orbit inception: the resultant decisions and consequences were formatted to allow direct comparison to the centralized method; though here rather than looking at correlations between position allocations, the resultant agent ΔV reserves and decision times are points of interest.

12.3.1 Array Formation

In instances of array formation, including initial dispersal and redistribution, it is clear that the distributed decision process implemented by auction is not capable of *reliably* forming an optimal decision in regards to minimizing a cost function across the agent community. Such a result highlights the nature of self-interested agents residing within a community: the 'greedy' agents are only concerned with minimization of their own expenses and this short-sighted nature results in a non-optimal result, even though a subset of the agents may result in an optimal placement.

Whilst not capable of reliably forming optimal decisions, as achieved by the centralized method, it is evident that the distributed auction method implemented can in some instances result in optimal assignment, as was observed for 7% of auction instances enacted for redistribution. Although not repeatedly optimal, the auction method is capable of making *sensible* decisions in all occasions: the penalty incurred through any non-optimal allocations is marginal, and the output response is certainly reasonable in that each agent follows a simple path, with minimum deviation from their dynamic state upon ejection from the mother-satellite.

It is not overly surprising that an altruistic auction is closer to optimal than that produced by a 'precedence' auction, since the precedence mechanism represents an additional action of greed within the agent community. However, in the auction enactments held, auction managers were randomly selected rather than being resultant from some form of higher level reasoning: whilst an agent will receive a payoff in the form of reduced ΔV expenditure for acting as an auction manager, this is at the expense of increased

communication and data processing, which will reduce the agent's ability to function in other areas.

Despite the promising results given by the auction methods implemented here, we are faced with the conclusion that to guarantee an optimal result for a small scale assignment problem, one should ultimately implement a centralized process. However, such centralized processes are computationally expensive, since central computation of all AAP permutations are required. Within Chapter 4 the use of a hybrid auction was introduced as a means of taking advantage of properties availed by both centralized and distributed processes: here we wish to take advantage of the guaranteed optimality of a centralized scheme with the computational efficiency of a distributed scheme. Such implementation of a hybrid scheme would fit within the investigated framework, with each agent being responsible for computing their own cost functions and relaying them to a central agent which then solves the assignment problem; a reliably optimal result is achieved with minimal computational overhead. The empirical evidence of this is observed within Figure 12.17, which illustrates the explosion in computation (time) requirements when performing a centralized decision compared to the same computation being carried out across the agent community[8]. The computational time depicted is that for complete solution using centralized or hybrid methods and is inclusive of cost matrix generation, for each agent within the community, and solution using the Hungarian method. It is clearly evident that the hybrid method offers superior performance in terms of computation times.

[8] All computation results are based on a Windows platform using an Intel Core 2 Duo 2.66GHz processor and 4GB of RAM

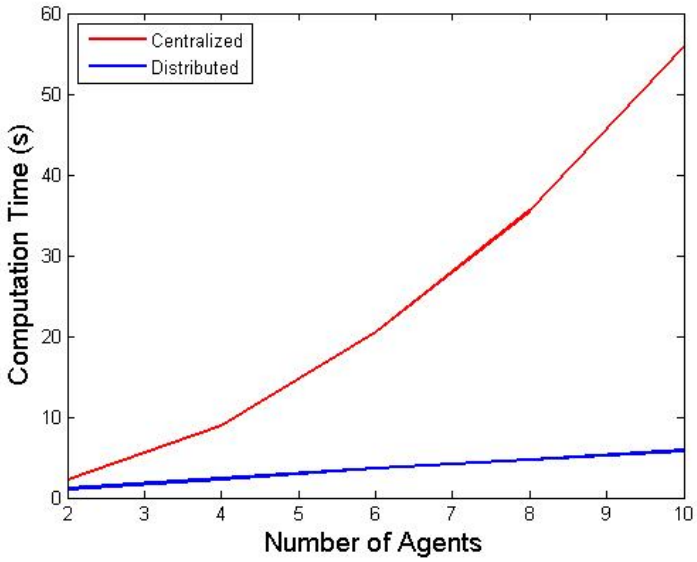


Fig. 12.17: Computational times to complete array point assignment using both centralized and hybrid processes: time is inclusive of cost matrix generation and solution through the Hungarian method.

12.3.2 Mission Maximization

In comparing an optimal method to auction methods for the purposes of maximizing agent community utility throughout the mission lifetime, it is evident that the auction methods are unable to produce the same level of optimality as a centralized method: alike the instance of agent position allocation, the greed of certain agents results in non-optimal point allocation and the shortsighted nature of the mechanism precludes self-sacrifice to permit a more desirable end result. Consequently, by implementing auction methods we result in agents exhausting their ΔV reserves; this does not happen in the same example mission being coordinated via centralized decision processes. However, whilst these traits do preclude certain aspects of optimality with regards to ΔV reserve maximization, other advantageous aspects as a result of an auction process are observed, which are not present within the optimal scheme.

By analysis of the graphs depicting standard deviation of agent fuel reserves as a consequence of AAP position allocations throughout the mission life, shown in Figures 12.11(a) and 12.11(b), we can observe a key aspect of an optimal method: the optimal method is concerned with the end result and not how this result is achieved. It is clear that the optimal method allows convergence to a very low standard deviation of agent ΔV reserves at the targeted mission end point; however mid-mission standard deviations are significantly greater and this is observed most clearly within 12.11(b). Comparing this to the auction methods, we can see that the standard deviations remain low throughout the mission duration; substantially lower than those for the optimal locations at the mission midpoint. Here we are observing the short-sighted and selfish nature of agent auctions providing the possibility of greater flexibility within the mission should an agent fail. In the instance of an agent failing mid-mission whilst implementing the centralized and optimal sequence of agent positioning, the large disparity in agent ΔV reserves presents the possibility of losing an agent with high ΔV reserves and this would detrimentally affect the remaining mission. Regardless of losing an agent with high utility, any agent loss would be in conjunction with the need to determine a new optimal agent position sequence for the remaining mission which, as is evident within Table 12.2.3, is computationally expensive: the centralized optimization process, relating to the investigated large scale problem, resulted in $12,488,169$ function evaluations with 396 iterations. Contrasting this to agent failure within a mission implementing auction methods, all agents retain a similar utility, and so there is no significance to loss of any agent other than degradation of u,v-plane coverage.

Considering agent losses observed through the implementation of auction based decisions, in all instances agent loss was suffered on the final orbit only: this was true for both the large and small scale problems investigated. The ΔV flow and the final agent

ΔV reserves, for both the small and large scale problems, are shown within Figures 12.12 to 12.16. Perhaps the most notable point here is that within the small scale problem, the altruistic agent auction process outperforms the Hungarian method: although this information is presented within Table 12.2.3, it is visibly evident within Figure 12.13(b). This is contrary to what one might expect, since the Hungarian method would naturally assign the most optimal position allocation upon each auction instance: however it can be seen here that iterative optimal placements do not necessarily result in achieving an optimal end result. Such differences in performance relating to the large scale problem were not witnessed: each auction mechanism failed to extend the complete agent community life to that achieved via the centralized method, resulting with loss of a single agent upon termination of the final orbit.

Observing the ΔV standard deviations resultant from implementing auction based decision methods, we can observe that there is maintenance of low ΔV standard deviations; perhaps the most interesting point of note is that within Figure 12.11(a) it is seen that the altruistic and Hungarian auctions result in virtually identical traces, indicating a near equivalence in their output decisions. The precedence auction, although retaining similar standard deviation magnitudes to the other auction mechanisms, exhibits a higher level of fluctuation in the trace: this can be attributed to the greed inherent with the method resulting in less optimal decisions being made.

One could ask questions pertaining to the validity of comparing auction methods with the presented optimal scheme, since one could foresee a centralized method being used to specify the action of maintaining equal ΔV reserves throughout the mission[9]. However both the centralized and distributed decision mechanism were intended to maximize system life and whilst it is true that the central (and hence optimal) method is superior in this instance, it has been observed that an *emergent* trait of implementing an auction method, in any form, is the maintenance of system flexibility in addition to increasing the system life beyond that possible with fixed agent positions. A non-trivial consideration relating to the advantages of an auction scheme is the time taken in forming a decision: it is clear that instances of auction methods require very little computation whilst an optimal solution is potentially prohibitive in terms of computation requirements.

12.4 Chapter Summary

This Chapter has examined the agent skill sets of control and decision making using both centralized and distributed methods.

[9] In fact this latter consideration has been achieved through application of the Hungarian method to the problem at each decision instance

The control method has been observed to provide the desired characteristics for the interferometry problem, permitting accurate position regulation and collision avoidance in the simulated scenarios of agent dispersal and reconfiguration, enabling the attainment of the desired interferometer configuration within the dynamics of the restricted three body problem. Here reference to 'accurate' is implicitly related to the accuracy of the filtering techniques implemented and analyzed within Chapter 9, which in turn are directly related to the accuracy of the devices being emulated.

We have observed that although not capable of forming optimal solutions, auction methods are capable of forming realistic and sensible decisions. Optimal centralized methods are concerned with the end result and not how this result was achieved: when considering single instances of AAP allocation this is indeed an advantage, however in a much larger scale problem this is seen to lead to disadvantages in both computation requirements and disparity of utility within the community. It is the opinion of this author that implementation of an optimal scheme for large scale problems comes at a price which is perhaps too high: agent auctions are not only computationally simplistic but offer advantages in large scale scenarios by maintenance of uniform agent utility throughout the mission duration, which in turn introduces an element of mission flexibility and superior performance in instances of agent failure. It has also been shown that a hybrid auction is capable of reliably forming optimal solutions in single instances of AAP assignment, without the computational overhead exhibited by a centralized scheme completing the same process: such evidence is clearly indicating that centralized decision methods applied to large groups are inefficient.

Considering the individual auction types implemented, it is clear that hybrid auctions for single decision instances represent optimality in terms of both result and computational efficiency; however such methods do not outperform other auction types in larger scale problems. It has been observed that an altruistic auction method can, in some instances, outperform the theoretically superior Hungarian method. The precedence auction, whilst impacting selection choice, represents a opportunity for agents to benefit through completing the position allocation computation: though here each agent was selected randomly, it was seen that the additional instance of agent greed was not overly detrimental to the community health[10]. All auction methods provide a rich infrastructure of collective actions regarding decision, planning and action wherein each agent shares responsibility for the success of the total mission. Such benefits are observed within any instance of computation and also with maintenance of near uniform agent community member utility (in this instance a near equivalence of ΔV resources) within the simulated

[10] It is noted and accepted that had a single agent acted as auction manager throughout the mission duration, the community health would have been negatively impacted.

mission.

So far this thesis has considered separated spacecraft interferometry in the context of a multi-agent system. Consideration has primarily been on key skills such as the ability to make and execute decisions relating to the desire of attaining a specific system configuration, all of which hinge on the extraction of relevant state information. Although significant, these issues constitute a very small proportion of the issues relating to the solution of PSSI in the MAS context. The following Chapter will present additional MAS issues, which although applicable to any MAS, will be formulated in the context of the exemplar PSSI mission.

13. CONSIDERATIONS FOR A MAS PSSI SYSTEM

Having presented and analyzed the agent skills sets envisaged for the PSSI application within Chapters 9 to 12, additional considerations pertinent to multi-agent system operation and design will now be presented. This Chapter will consider issues such as the system classification and resulting agent cycle, in addition to generalized design considerations for MAS which have become evident during the course of the work undertaken within this thesis.

13.1 PSSI System Classification

The agent system classification for the PSSI system may be readily formulated into that of a BDI[1] model. As detailed within Chapter 2, numerous BDI programming languages exist: although all these languages cling to the BDI concept, there is no strict agreement between notions or definitions within the respective languages relating to what constitutes a desire, belief or intention. Such discrepancies result in difficultly in applying differing programming languages to solve a particular problem. Due to this an agent infrastructure layer (AIL) is being developed by Liverpool University which aims to act as an intermediary layer by providing a common semantic basis for BDI languages [168].

AIL uses first order literals, which are propositional variables[2] for the representation of beliefs, goals and actions. The AIL belief base is composed of a set of belief formulae into which a Prolog[3] reasoning engine is incorporated: a formula is *believed* if it is a formula within the set. Desires (or equivalently goals) within BDI languages are somewhat vague in that the semantics and terminology is generally quite subtle: within [169], four separate types are referenced relating to achieve, perform, maintain and query. Within AIL the notion of a deed stack is used to prescribe a sequence of tasks required to attain

[1] BDI: Beliefs, Desires and Intentions model, as presented within Chapter 2.

[2] A propositional variable, also called a sentential variable or sentential letter, is a variable which can either be true or false.

[3] Prolog is a declarative logic programming language expressed in terms of relations. Execution is triggered by running queries over these relations, wherein the Prolog engine attempts to find a resolution refutation of the negated query.

a particular goal. Such formulation permits differentiation between high level goals and methods to achieve such goals, whilst still allowing outstanding goals to be identified and committed to. As with all agent programming language development, an action is something performed by an agent within the environment they inhabit and it is these actions which modify the agent's belief base. An action can be viewed as the execution of a plan: indeed within all BDI languages plans are a fundamental aspect. Within AIL, the terminology of *plan* is applied to methods for the solution of a particular problem, as specified by the agent designer, and the deed stack represents the set of pending actions by the agent to attain a goal. It should be noted that within AIL, goals encompass the notions of desire and intent; actions explicitly relate to agent output.

Of consideration now is how to formulate the PSSI system such that it may fit into an existing agent architecture for possible implementation. For the sake of clarity in presentation, component BDI's will not be reduced to literals but will be presented in the format of high level abstract concepts.

Individual Agent Beliefs The agent belief base is constructed based upon agent percepts, ultimately transformed into literals used to construct the belief formulae. The abstract concepts required to formulate these belief formulae may be composed of the following entities:

1. Current complete agent state, as presented within Chapter 9. Internally this requires knowledge of the current agent tasked with the NLS data fusion used within the translational Kalman filter.
2. Desired state, relating to (if relevant) allocated array position and orientation.
3. Agent (own) health[4].
4. Agent community health.
5. Existence of an optimal array relating to the current number of agents within the community.
6. Ability to achieve a desire. This belief is achievable through formal logic reasoning based upon current beliefs, in particular an event list in conjunction with a skill set (or plan set) which may be able to bring about a certain change. This is provided for within AIL through the Prolog reasoning engine.

Within the agent belief base exists a special form of belief, which in a similar manner to AgentSpeak, we shall refer to as *events* and these relate to triggering beliefs.

[4] Here health status corresponds to individual agent health and consequently the ability for an agent to attain desires. Health encapsulates fuel, power and system components pertinent to mission success.

A triggering belief is a significant percept permitting the *possible* achievement of a desire and exists as a precondition used within the Prolog reasoning. For the PSSI scenario, an event list will contain:

1. End of agent hibernation. This represents a somewhat trivial event at which point the agent is activated.
2. Agent ejected from transport vessel (the mother-satellite).
3. Agent failure within the community (fellow agent, total loss).
4. Partial systems failure of agent resulting in degraded performance.
5. Catastrophic systems failure resulting in loss of agent functionality.

Individual Agent Goals The set of agent goals, which in the AIL format encompass both desires and intentions, relate to the required agent actions for the optimal performance of a PSSI array and these may or may not be conflicting[5]. The set includes:

1. Maintain components of belief base.
2. Keep agent community informed of current health status.
3. Be located in and maintain an array position, with the correct state.
4. Maximize the mission performance. This desire has a further subset consisting of minimization of own fuel consumption and maintaining an optimal array corresponding to the number of 'healthy' agents within the community.
5. Harvest data for the purposes of interferometry.

As is evident each desire has a related set of pre-conditions which must be satisfied prior to the ability of the goal to be attained: this relates directly to triggering events and the deed stack within AIL. The agent deed stack, composed of pending intentions, is constructed based upon the pre-specified plans as formulated by the agent designer: these plans have been termed *skills* within this thesis. Within BDI languages, plans are triggered according to the agent's belief base and are of the form $(trigger, guard, body)$ wherein trigger relates to a triggering event, guard is the set of literals required to be valid for the plan to be applicable and body is that to be placed onto the current deed stack.

[5] An instance of conflicting actions in this scenario relates to the desires of an agent to minimize fuel expenditure yet attain an array location, which intrinsically requires the expenditure of propellant reserves.

Individual Agent Actions Actions relate directly to the agent output and within the PSSI scenario this corresponds to the execution of agent skills. It is these actions which will influence the agent percepts, leading to the ability of an agent to determine if a particular goal has been achieved or not and the possible requirements to switch goals being pursued.

Within [169] an informal representation of the AIL reasoning cycle is given and replicated within Figure 13.1. The cycle commences at node **A** and progresses to node **F**, upon which time the cycle repeats. Initially an intention, denoted by i and with it's associated deed stack, is selected and we are transferred to node **B**. In a similar fashion to the PRS architecture presented in Chapter 2 a set of applicable plans, denoted by P, are generated based upon the agent's belief base and plan library (skill set) where the process transfers to node **C**. At this point a single plan, p, is selected and the associated deed stack for this single plan is concatenated with the current deed stack, bringing the process to node **D**. The upper-most deed is handled in the appropriate manner which leads to the (possible) creation of new intentions and eventual action (node **E**). Perception occurs between nodes **E** and **F**, and these percepts are used to update the current belief base and posting of new events: the reasoning cycle repeats.

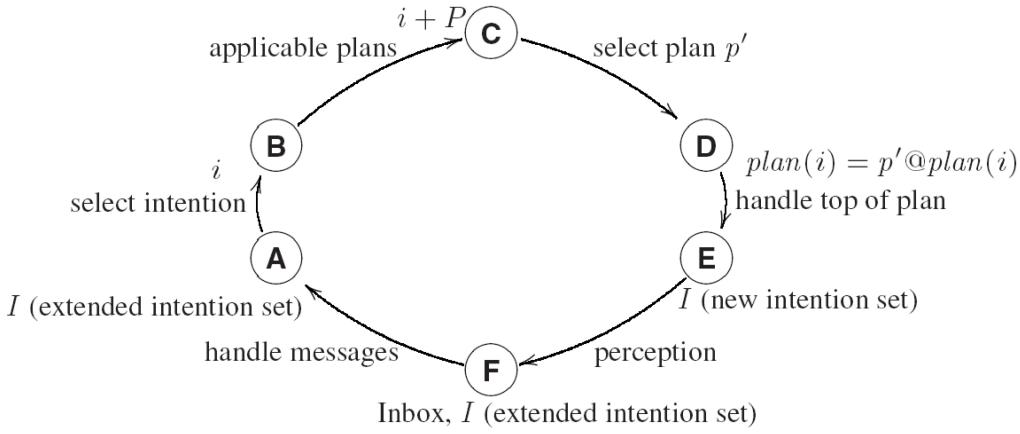


Fig. 13.1: An informal representation of the AIL reasoning cycle.

13.1.1 Analogies to Hybrid Control Theory

One cannot help but draw a parallel between the triggered plan mechanism within BDI languages, mentioned within the preceding Section and denoted by $(trigger, guard, body)$, with that of hybrid system control implementation. Indeed, hybrid control systems also share the formal verification properties of BDI languages in that it possible to investigate reachability of certain states [170, 171].

A hybrid system is a system of interacting continuous and discrete components: these components interact at *event* or *trigger* times, when the continuous state reaches certain prescribed sets in the continuous state space. A hybrid dynamical system is an indexed collection of dynamical systems with a map to switch between these systems given that the current state satisfies certain conditions, prescribed by its membership in a specified subset of the state space. One could envisage control operation flow in such an environment being prescribed by $(guard, body)$, wherein the *guard* is used to enable the control *body* dependant upon location within the state space.

Whilst there are similarities within both BDI agent implementation and hybrid systems control, one should not forget the key difference and that which relates back to our separation of software agents and software objects within Chapter 2: *'Objects do it for free; agents do it because they want to'*. In a similar fashion, agents will perform reachability analysis as part of a *desire* to attain a specified objective and actions are triggered through the perception of opportunity. In contrast, by their definition, hybrid control systems are automatons: carrying out a sequence of operations as specified within their data structure. Whilst some may view such anthropomorphisms as needless, it is nevertheless the mantra to which the multi-agent community abides.

13.2 Consistency Within Skill Development

The need for consistency is prevalent throughout agent system development: in addition to the inherent need of consistency within agent communications, the development of AIL was forced through the need to provide a common interface for BDI agent programming languages and hence readily avail the developer with a suite of tools at their disposal. The application of AIL is certainly of merit and goes a long way in solving the development phase which an MAS programmer will face. Another, potentially more tricky situation, arises within the provision of skill sets to an agent. At present, skills sets (or plans), are developed on a per-platform basis, using whichever language the particular developer is comfortable with, to provide a one-off solution which is implementable only within their particular solution. The skill sets developed and presented within Chapters 9, 10 and 11 are prime examples of this: MATLAB was used throughout as the programming

and simulation language; the routines as they stand are not directly implementable in any other package. By necessity, all knowledge pertinent to these routines was written in a (hopefully) clear and concise fashion such that any reader would be able to replicate the methods should they desire. This has been the basis of human knowledge catalogue and transfer since the advent of written communication.

It is clear that there exists an immense collection of human knowledge which is both contained and potentially lost within research papers. Let us consider the typical life cycle of a journal paper whose contents are to be used to provide an agent with a particular skill: a researcher performs some form of work which results in a theoretical publication; the paper is shelved for later reference by a future reader, for which the subject matter might be of interest; an interested reader locates the paper, extracts the basic principles contained and replicates these within their own code; this code is then transferred to an agent for implementation in a problem for which the skill, replicated by the reader, may help to solve.

This process is certainly typical of an early stage research process, but let us consider the key flaw within this method: time is wasted by the reader in replicating existing code produced by the original author, who is the specialist regarding the documented techniques[6]. One might form certain questions regarding this:

1. What if the code was directly available for use?
2. What if the code was directly available for use in the required language?
3. What if an agent could access this code directly and subsequently use it?
4. What if an agent could read a published journal paper and immediately utilize the knowledge contained within?
5. What if a human *or* agent could read the published paper and immediately utilize the knowledge contained within?

It is becoming increasingly common that code relating to particular research fields is made freely available online by the author: hence our first question is theoretically answered but immediately presents the second question. In a fortunate scenario (for the reader), the code is written in the language which is to be implemented; otherwise the reader is again faced with the need to either interface differing languages or translate the code into that required for the intended application. The final three questions are similar and

[6] The replicated code may well contain bugs, which at best will delay progress and at worse will result in erroneous results.

build upon the concept of an agent being able to read a paper for itself and immediately be able to implement the knowledge contained therein. Regardless of the requirements to achieve this, such a possibility is certainly advantageous and completely eliminates the key problem highlighted previously relating to wasting valuable research time in repeating existing work. In concept, utilization of the knowledge contained within a journal paper would be as simple as a user invoking particular routines within a program, executing in a single native language; much alike a MATLAB toolbox. This is a possibility through the use of system English (sEnglish).

sEnglish is a natural language programming method which, having specified the internal ontology and written related commands in a user specified natural language, and compiles the structured sentences into MATLAB code for implementation. In addition, sEnglish can produce a pdf document detailing the ontology (conceptual structures) used within the coding and the processes completed by the code, as specified by the natural language program written by the user. The output is a latex and HTML PDF document which a human may read to understand the implementation of a particular piece of code: the document is essentially the code and as such this document (in electronic format) may be read by an agent and directly implemented. Consequently we are presented with the possibility of publishing knowledge for *both* humans and machines, where humans or machines may be able to read the same document and implement the knowledge contained.

Through the use of sEnglish, a database of agent skills could be created and made publicly available to both humans and agents: one can foresee that initially a human would search for a paper relevant to the requirements of an agent, later to be replaced by an agent searching for a relevant paper to a current problem faced.

Generic examples of sEnglish papers are available online at [172] and the skill of predicting controlled agent motion in a 6DoF environment, through a continuous time sliding mode control regime[7] and written by this author, within [173].

The benefits of formulating a standardized set of skills for direct use by the agent community would not be too dissimilar from the benefits experience by the computer vision community through the OpenCV library[8] and the agent community through existing standards as provided by FIPA[9]. Such libraries of pooled and documented knowledge, to

[7] It was this skill which was used within agent cost function development, as presented within Chapter 11 and implemented within Chapter 12.

[8] The Open Computer Vision (OpenCV) community provides an open source library of documented C and C++ routines mainly aimed at real time computer vision tasks; help is provided through a Wiki based community [174].

[9] The Foundation of Intelligent Physical Agents (FIPA) is a non-profit IEEE Computer Society standards organization that promotes agent-based technology and the inter-operability of its standards with other technologies [175].

which access is immediate, permit rapid development and implementation; which is the key to successful research.

13.3 Chapter Summary

This Chapter was intended to convey the huge breadth of considerations when designing and implementing a multi-agent system and in doing so has highlighted that perhaps the greatest consideration is the need for consistency and standards; FIPA is an organization dedicated solely to the formulation of standards within agent systems, though universal adoption of these standards within industry is not evident. AIL presents an agent formulation methodology enabling the implementation of various agent programming languages. Whilst the possibility of sEnglish to provide a standard method of formulating knowledge accessible to both humans and agents cannot be highlighted enough, perhaps its greatest strength is the ontological development that allows for direct integration within a true agent system.

14. CONCLUSIONS & FUTURE WORK

This thesis has presented an analysis of physically separated spacecraft interferometry in the context of a multi-agent system stationed in a libration point orbit. The main focus has been on the development of key agent skills, including state estimation, guidance, control and decision methods to attain the desired system output. Within the consideration of decision methods, a comparison between centralized and distributed decisions was made. Whilst effort was primarily placed in the development of these skills, additional considerations pertinent to the system development, such as the resultant agent cycle, were also discussed.

The discrete time control method integrated Kalman filtering with sliding mode control, using potential function guidance to achieve velocity and attitude tracking. In the presented scenario, interest was placed on autonomous controlled motion upon ejection from a mother satellite to enable the agents to be controlled accurately to specific locations and orientations in order to form an interferometer array and reconfiguration of the array to compensate for failed agents or to maximize the mission duration. Although seemingly a niche application, the methods are equally valid applied to any other vehicular agent system such as UAVs or AUVs when consideration is based upon inter-agent regulation.

In formulating and comparing decision methods we have seen that whilst central methods are superior for achieving a globally optimal result, in cases of large scale problems, central methods result in loss of agent system versatility and resilience in the presence of agent failure. The auction methods presented, whilst not capable of performing optimal decisions, invariably make sensible decisions and act to maintain system versatility with minimal computational overhead: permitting superior results in non ideal world scenarios. It is the choice of a system designer as to whether such a trade off is beneficial for a mission, and in some instances perhaps centralized methods are advantageous; however in the opinion of this author, the benefits of auction methods far outweigh any advantages in optimality availed by centralized methods. Indeed, the auction methods implemented highlight clearly the advantages sought by multi-agent system developers in terms of versatility and minimal loss of utility in the presence of component failure; this increase in versatility is also combined with greater autonomy within the system.

The final topic considered within this thesis fell upon the somewhat obvious need for consistency within multi-agent system development. Although this is a rather obvious statement and numerous organizations exist with the sole intention of promoting standards[1], there are serious disparities within methodologies stemming largely from institutional backgrounds, resulting in favored programming languages or indeed the agent system theory being implemented. We are also faced with a wealth of knowledge directly applicable to numerous multi-agent systems being lost within literature, a problem which could potentially be overcome through adopting standards within publication such that papers are machine readable.

14.1 Future Work

In terms of integrating agent skill sets to complete the intended mission scenario, within simulation we have seemingly achieved a working model: each agent initializes and runs their respective Kalman filters to obtain state information, this information is used within the control that is applied according to system decisions and the resultant guidance for each agent. Instances of controlled motion for agent ejection and reconfiguration have been simulated and proven effective, permitting sensor bounded accuracy in position tracking and inter-agent collision avoidance. Resultant decisions are seamlessly executed, though it should be noted that due to the time restrictions involved with optimization of mission duration via a centralized scheme, these decisions were made off-line and introduced to the simulation environment; decisions made via auction methods were implemented online during simulation.

The work presented within this thesis is not without its limitations and there is a large scope for improvement and expansion of the areas investigated. These two areas shall now be considered.

System Improvement Whilst a seemingly complete simulation model, there are nevertheless limitations present as a consequence of running all experimentation within simulation: all agent processes were completed in a single MATLAB instance. Whilst this is not necessarily problematic with regards to the simulation dynamics[2], we are faced with the fact that all agent communications were also completed within

[1] A prime example here is the Foundation for Intelligent Physical Agents (FIPA) [175].

[2] Within the simulation environment of the RTBP, numerous modeling simplifications are made, as detailed within Chapter 6, to produce an elegant and workable solution: the real dynamics of the halo orbit do not exhibit such restrictions. Although a pertinent consideration, such discrepancies applied to an agent system is unlikely to cause any divergence from the modeled behavior, since the entire system would be equally affected by any unmodeled disturbances.

the MATLAB environment. It was assumed that these communications were instant and complete, which in a real scenario is a large assumption: real systems are subject to communication lag, noise, bit loss and the possibility of communication drop-out. None of these communication issues were introduced into the interferometry scenario and this represents an immediate position where the results presented within this thesis could be made more concrete.

Limitations within simulation may be avoided through the implementation of both control and communication routines, on hardware, in a ground based environment. Very few environments suitable for these purposes exist, though the author has been involved with development and construction of a 5DOF testing facility at the University of Southampton, depicted in Figure 14.1, which is the subject of continued work. Whilst it was not the intention of this thesis to cover ground based hardware-in-the-loop testing, it is noted that such testing methods greatly extend any theoretical and computationally simulated results.

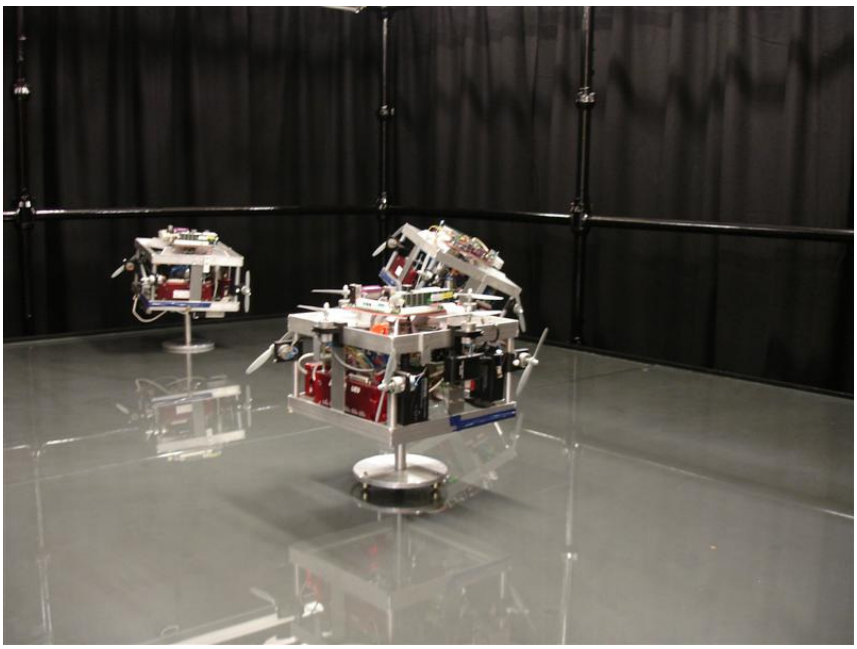


Fig. 14.1: Ground based 5DOF testing facility for guidance, navigation, control and communication routines, based at the University of Southampton.

Indeed, aside from the introduction of additional system constraints to the simulation or perhaps implementation within hardware, there are also methods implemented which, should the work be repeated, would be completed differently. The

guidance method used is based upon potential fields, which although having a substantial history in the field of autonomous vehicles, is hindered by lack of verifiability. Although proven effective in the scenarios considered within this thesis, we are still faced with the fact that the potential function methods implemented within the controller prevent us for proving that the agents will reach their intended locations and not be stationed at local minima. Such problems could be overcome using a more powerful, distributed agent planning mechanism: the aim here would not be to form a centralized plan for the agents to execute, but for each agent to plan their own route and modify this route should they detect the possibility of a collision[3]. Positioning information for each agent is based upon a coordinate frame attached to a central agent, and the Kalman filter methods for both location and translational velocity state estimation are given within this frame using centralized computation. An advance in this would be removal of the centralized method, such that the agents are tasked with maintaining a lattice formation, which would require maintenance of distances from nominated agents. Such rigidity and persistence of directed graphs has been investigated within [176], though the far more complex issue of moving $3D$ formations subject to errors within position measurement has not been treated.

System Expansion There are two main extensions to the investigated subject, which sadly were not touched upon due to time restrictions. The first of these relate to the reasoning behind election of an auction manager: in all auction scenarios the auction manager was selected at random. Whilst such a method was entirely valid for the purposes of investigating auction based decisions, a more complete scenario would have included some form of electoral process such that an auction manager is actively sought (or conversely the position of auction manager is actively sought). Whilst it could be anticipated that this would call upon the introduction of a more realistic simulation of agent communications and methods to determine current workloads with potential payoff, such conjecture is perhaps best omitted.

An additional extension relates to the task of extending the mission duration through phased array point assignment. Within the presented work, array point reassignment was decided upon each orbit completion; such sparse decision instances could certainly be improved upon. Implementing a centralized process, it is entirely possible to divide the decision instances into fractions of an orbit and solve this using the presented binary integer solution, though this will be at the expense of huge

[3] This would entail some form of forward prediction mechanism and a library of contingency plans, varying from 'slow down' to 'evade'.

computation requirements. Extension of the presented auction methods, applied to ad-hoc phased array point assignment, wherein the agents themselves are responsible for selecting instances at which time they should form decisions, would be of great interest. This would involve the introduction of increased rule sets to the agents such as to modify their behaviors; whilst it is trivial to place conjecture on how this would operate, a prime concern would be that of preventing cyclic motion within the agent system.

14.2 Summary

It is hoped that the content of this thesis has conveyed the applicability of multi-agent systems to real engineering problems and how such methods can not only solve the problem, but offer additional benefits unobtainable from more traditional, centralized solutions. Whilst the formation of a discrete controller is one aspect of this thesis which the author hopes was of interest with regards to development and the obvious applicability in other agent vehicle areas, it is also hoped that the discourse of centralized, decentralized and hybrid decision making was equally of interest. The multi-agent system community champions the use of decentralized methods over their centralized counterpart: usually without any formal proof. Within this thesis it has been shown that centralized methods do have their place and indeed outperform the decentralized counterpart on every occasion; however certain applications of centrality should certainly be limited. For instances of discrete decision making, which in the presented scenario included initial dispersal and instances of reconfiguration due to failed agents, it is more efficient to use a centralized method: the communications involved are no more than that required by pure auction methods and a guaranteed optimal result will be achieved. Whilst it is equally possible to implement a pure auction using any of the developed mechanisms at minimal cost to the global result, in doing so there is no real benefit observed, except enforcing a purely decentralized decision process. Instances of larger scale centralized solutions have been shown to be, although unarguably superior in terms of optimality, detrimental to system versatility. In seeking an optimal end result, centralized methods are ignorant of the interim eventualities; the converse is true for auction methods. Implementation of auction methods, in the instances investigated here, are unlikely to produce an optimal result as this is a statistical improbability. However, auction methods will avail the system community with far more mid-term benefits: in the scenario presented, these benefits manifested themselves in terms of increased tolerance to agent failure and minimization of computational overheads. It could be argued that, in the presented scenario, a centralized process

would have an entire orbit period to complete the necessary optimization[4] and so such time constraints are not an issue. However, not only is the optimization resource consuming, which would impact upon other system functionality, but these auction mechanisms are not restricted to application within an interferometer scenario. Looking back into the presented space based multi-agent systems within Chapter 2, for which all cost functions and control requirements are valid, we are presented with numerous scenarios where timeliness of decisions is an over-riding factor and critical so as not to miss opportunistic events[5]. Perhaps the most pertinent example here is the conceptual APIES or ANTS program, presented within Chapter 2, wherein timely decisions must be made regarding asteroid belt exploration.

This thesis has presented solutions to key aspects of the spacecraft interferometry problem: those of state determination, discrete time control control and forming appropriate decisions to envisaged scenarios; decision mechanisms were generated and investigated using both centralized and distributed methods. Within the investigation of centralized and distributed decision methods it was observed that applications of multi-agent system principles, those of distribution, offer advantages over more traditional centralized methods. These advantages manifested themselves in the ability to maximize utility of the array with minimal computational overhead; vindicating many of the claims made by the multi-agent systems community regarding benefits of applying distributed techniques to complex systems.

[4] For the ISEE-3 mission orbit replicated, this would correspond to 177 days.

[5] Issues of collision avoidance are removed from timeliness considerations since such requirements are central to the guidance mechanism and this runs separately to any form of high level system decisions.

BIBLIOGRAPHY

[1] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley and Sons, LTD, 2002.

[2] M. Wooldridge and N.R. Jennings. Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.

[3] N. R. Jennings and M. Wooldridge. *Applications of Intelligent Agents*, pages 3–28. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998.

[4] L. Padgham and M. Winikoff. *Developing Intelligent Agent Systems, A Prctical Guide*. John Wiley and Sons, LTD, 2005.

[5] J.M. Vidal. *Fundamentals of Multiagent Systems: Using NetLogo Models*. Unpublished, 2006. `http://www.multiagent.com/fmas`.

[6] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2003.

[7] P. Maes. Artificial Life Meets Entertainment: Lifelike Autonomous Agents. *Commun. ACM*, 38(11):108–114, 1995.

[8] R.H. Bordini, J.F. Hubner, and M. Wooldridge. *Programming Multi-Agent Systems in Agent Speak Using Jason*. John Wiley and Sons LTD, 2007.

[9] F. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems With Jade*. John Wiley and Sons LTD, 2007.

[10] J. Liu. *Autonomous Agents and Multi-Agent Systems*. World Scientific, 2001.

[11] V.Tan, D.Yoo, and M. Yi. A Multiagent-System Framework for Hierarchical Control and Monitoring of Complex Process Control Systems. In *PRIMA* 08*: Proceedings of the* 11*th Pacific Rim International Conference on Multi-Agents*, pages 381–388, Hanoi, Vietnam, 2008.

[12] S.M. Veres and J.Luo. A Class of BDI Agent Architectures for Autonomous Control. In *Proceedings of the* 43^{rd} *IEEE Conference on Decision and Control, CDC* 2004, pages 4746–4751, Atlantis, Paradise Island, Bahamas, December 2004.

[13] H.V.D. Parunak. Go to the Ant: Engineering Principles from Natural Multi-Agent Systems. *Annals of Opertation Research*, 75(1):69–101, 1997.

[14] H. Barringer, M. Fisher, D.M. Gabbay, G. Gough, and R. Owens. METATEM: A Framework for Programming in Temporal Logic. In *Stepwise Refinement of Distributed Systems, Models, Formalisms, Correctness, REX Workshop*, pages 94–129, London, UK, 1990.

[15] M. Fisher. Concurrent METATEM - A Language for Modelling Reactive Systems. In *PARLE '93: Proceedings of the 5th International PARLE Conference on Parallel Architectures and Languages Europe*, pages 185–196, London, UK, 1993.

[16] M. Fisher. A Survey of Concurrent METATEM - the Language and its Applications. In *ICTL '94: Proceedings of the First International Conference on Temporal Logic*, pages 480–505, London, UK, 1994.

[17] M.E. Bratman. *Intention, Plans, and Practical Reason*. CLSI Publications, 1999.

[18] M.J. Mataric. Using Communication to Reduce Locality in Distributed Multi-Agent Learning. *Journal of Experimental and Theoretical Artificial Intelligence*, 10:357–369, 1998.

[19] G. Dudek, M.R.M. Jenkin, and D. Wilkes. A Taxonomy for Multi-Agent Robotics. *Autonomous Robots*, 3:375–397, 1996.

[20] M. Dorigo and L.M. Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, 1:53–66, 1997.

[21] K. Vittori, G. Talbot, J. Gautrais, V. Fourcassié, A.F.R Araujo, and G. Theraulaz. Path Efficiency of Ant Foraging Trails in an Artificial Network. *Journal of Theoretical Biology*, 239:507–515, 2006.

[22] A. Drogoul. When Ants Play Chess (or can strategies emerge from tactical behaviors). In *In Proceedings of Fifth European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW* 93, pages 13–27. Springer, 1995.

[23] S. Wagner, M. Affenzeller, and I.K. Ibrahim. Agent-based Problem Solving: The Ant Colonies Metaphor. In *Fifth International Conference on Information and Web-based Applications and Services (iiWAS)*, volume 170, Jakarta, Indonesia, 2003.

[24] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26:29–41, 1996.

[25] R. Geetha-Ramani, P. Viswanath, and B. Arjun. Ant Intelligence in Robotic Soccer. *International Journal of Advanced Robotic Systems*, 5(1):49–58, 2008.

[26] H.V.D. Parunak. Practical and Industrial Applications of Agent-Based Systems. Technical report, The Industrial Technology Institution, 1998.

[27] M. Ljungberg and A. Lucas. The OASIS Air Traffic Management System. In *In Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, Soeul, Korea, 1992.

[28] B. Burmeister, A. Haddadi, and G. Matylis. Application of Multi-agent Systems in Traffic and Transportation. In *In Proceedings of Software Engineering*, volume 144, pages 51–60, 1997.

[29] N.V. Findler and R. Lo. An Examination of Distributed Planning in the World of Air Traffic Control. *Journal of Parallel Distributed Computing*, 3(3), 1986.

[30] K. Tumer and A. Agogino. Distributed Agent-based Air Traffic Flow Management. In *AAMAS '07: Proceedings of the 6th International Joint Conference On Autonomous Agents And Multiagent Systems*, Honolulu, Hawaii, 2007.

[31] Incident News Web Article. `www.incidentnews.gov/incident/$7733$`.

[32] Kiva Systems. `http://www.kivasystems.com`.

[33] Retail Info Systems News. `www.risnews.com`, September 2008.

[34] N. Muscettola, P.P. Nayak, B. Pell, and B.C. Williams. Remote Agent: To Boldly Go Where No AI System Has Gone Before. *Artificial Intelligence*, 103:5–47, 1998.

[35] D.E. Bernard, G.A. Dorais, C. Fry, E.B. Gamble, B. Kanefsky, J. Kurien, W. Millar, N. Muscettola, U. Nayak, B. Pell, K. Rajan, and N. Rouquette. Design of the Remote Agent Experiment for Spacecraft Autonomy. In *In Proceedings of the* 1998 *IEEE Aerospace Conference*, pages 259–281, 1998.

[36] NASA. Remote Agent. `http://ti.arc.nasa.gov/projects/remote-agent`, 1999.

[37] J.B. Mueller, D.M. Surka, and B. Udrea. Agent-based Control of Multiple Satellite Formation Flying. In *Proceedings of the* 6*th International Symposium on Artificial Intelligence and Robotics and Automation in Space: i-SAIRAS*, pages 385–389, Montreal, Canada, June 2001.

[38] K. Thanapalan and S.M. Veres. Agent Based Controller for Satellite Formation Flying. In *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference*, pages 385–389, Melbourne, Australia, December 2005.

[39] S. Chien, R. Sherwood, G. Rabideau, R. Castano, A. Davies, M. Burl, R. Knight, T. Stough, J. Roden, P. Zetocha, R. Wainright, P. Klupar, J.V. Gaasbeck, P. Cappealeare, and D. Oswald. The Techsat-21 Autonomous Space Science Agent. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems: AAMAS*, Bologna, Italy, July 2002.

[40] Space News Article. `www.space.com/spacenews/archive$06$/formation_$1204$.html`.

[41] ESA. Swarm - The Earths Magnetic Field and Environment Explorers. Technical report, European Space Agency, April 2004. ESA SP-1279(6), Reports for Mission Selection: The Six Candidate Earth Explorer Missions.

[42] ESA Swarm Mission. `http://www.esa.int`.

[43] Surrey Satellite Technology Limited. `www.sstl.co.uk`.

[44] Mini AERCam. `http://aercam.jsc.nasa.gov`.

[45] CubeSat. `http://www.cubesat.org`.

[46] e-CUBES: 3D Integrated Micr/Nano Modules for Easily Adapted Applications. `http://ecubes.epfl.ch/public`.

[47] E. Geneste and D. Barnes. Aerobot Airdata Measurement for Planetary Exploration. Technical report, Manchester University, 2001. `http://users.aber.ac.uk/dpb/timr_$01$.pdf`.

[48] S. Dubowsky, K. Iagnemma, S. Liberatore, D.M. Lambeth, J.S. Plante, and P.J Boston. A Concept Mission: Microbots for Large-Scale Planetary Surface and Subsurface Exploration. In *In Proceedings of the Space Technology and Applications Inteligence Forum*, pages 1449–1458, 2005.

[49] P. DArrigo. The APIES Mission Executive Summary for Publication on ESA Web Pages. Technical report, Astrium Ltd, 2004.

[50] W. Shen, P. Will, and B. Khoshnevis. Self-Assembly in Space via Self-Reconfigurable Robots. In *Proceedings of the* 2003 *IEEE International Conference On Robotics & Automation*, Taipei, Taiwan, September 2003.

[51] D. Izzo and L. Pettazzi. Self Assembly of Large Structures in Space Using Intersatellite Coulomb Forces. In 57*th International Astronautical Congress*, Valencia, Spain, 2006.

[52] D. Izzo, L. Pettazzi, and M. Ayre. Mission Concept for Autonomous on Orbit Assembly of a Large Reflector in Space. In 56*th International Astronautical Congress*, Fukuoka, Japan, 2005.

[53] I. Becky. Phase I Study: Extreemely Large Swarm Array Of PicoSats For Microwave / RF Earth Sensing, Radiometry and Mapping. Technical report, NASA Institute of Advanced Concepts (NIAC), April 2005. `http://www.niac.usra.edu/files/studies/final_report/$942$Bekey.pdf`.

[54] ESA. ESA's Water Mission: SMOS. Technical report, European Space Agency, June 2004. BR-224.

[55] P. Jantapremjit and D. Austin. Design of a Modular Self-Recongurable Robot. In *In Proceedings of Australian Conference on Robotics and Automation*, 2001.

[56] M. Yim, K. Roufas, D.D., Y. Zhang, C.Eldershaw, and S. Homans. Modular Reconfigurable Robots in Space Applications. *Autonomous Robots*, 14(2-3):225–237, 2003.

[57] M. Yim, W.M Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G.S. Chirikjian. Modular Self Reconfigurable Robotic Systems, Grand Challenges and Future Opportunities. IEEE Robotics and Automation Magazine, March 2007.

[58] Seth Copen Goldstein and Todd C. Mowry. Claytronics: A Scalable Basis for Future Robots. In *RoboSphere* 2004, Moffett Field, CA, November 2004.

[59] S. Nakasuka, Y. Sugawara, H. Sahara, K.Koyama, T. Okada, and C. Kobayashi. Panel Extension Satellite (PETSAT) - A Novel Satellite Concept Consisting of Modular, Functional and Plug-in Panels. In K. Fletcher, editor, *First Workshop on Innovative System Concepts*, volume 633 of *ESA Special Publication*, pages 81–88, August 2006.

[60] O. Brown and P. Eremenko. Fractionated Space Architectures: A Vision for Responsive Space. In *Fourth Responsive Space Conference*, Los Angeles, CA, April 2006.

[61] A.B. Meinel and M.P. Meinel. Optical Phased Array Configuration for an Extremely Large Telescope. *Journal of Applied Optics*, 43(3):601–607, 2004.

[62] J.D. Monnier. Optical interferometry in astronomy. *Reports on Prgress in Physics*, 66(1):789–857, 2003.

[63] J.D. Monnier. Optical Interferometry in Astronomy. In *Reports of Progress in Physics*, volume 66, pages 789–857, 2003.

[64] S.T. Ridgeway. Solar Optical Interferometry. In *Infrared Solar Physics*, volume 154 of *IAU Symposium*, pages 567–578, 1994.

[65] P.R. Lawson. *Principles of Long Baseline Stellar Interferometry*. NASA JPL, 2000. Course notes from the 1999 Michelson Summer School, held August 15 – 19, 1999.

[66] B.F. Burke and F. Graham-Smith. *An Introduction to Radio Astronomy*. Cambridge University Press, 2002.

[67] R.A. Perley, F.R. Schwab, and A.H Bridle. *Synthesis Imaging in Radio Astronomy: A Collection of Lectures from the Third NRAO Synthesis Imaging Summer School*. Astronomical Society of the Pacific Conference Series, 1989.

[68] M. Felli and R.E. Spencer. *Very Long Baseline Interferometry: Techniques and Applications (NATO Science Series: C)*. Kluwer Academic Publishers, 1989.

[69] N.J. Miller, M.P. Dierking, and B.D. Duncan. Optical Sparse Aperture Imaging. *Journal Optical Society America*, 46(23):5933–5943, 2007.

[70] J.R. Fienup. MTF and Integration Time Versus Fill Factor for Sparse-Aperture Imaging Systems. In *Imaging Technology and Telescopes, Proceedings SPIE*, San Diego, CA, July 2000.

[71] M.J.E. Golay. Point Arrays Having Compact, Nonredundant Autocorrelations. *Journal Optical Society America*, 61:272–273, 1971.

[72] T.J. Cornwell. A Novel Principle for Optimization of the Instantaneous Fourier Plane Coverage of Correlation Arrays. *IEEE Transactions on Antennas and Propogation*, 36(8):1165–1167, August 1988.

[73] A.B. Meinel and M.P. Meinel. Large Sparse Aperture Space Optical Systems. *Optical Engineering*, 41:1983–1994, August 2002.

[74] L.E. Kopilovich. Optimization of Non-redundant Apertures on Integer Grids. In *Physics and Engineering of Millimeter and Submillimeter Waves, Third International Kharkov Symposium*, volume 2, pages 572–574, September 1998.

[75] M. Lou. Development and Application of Space Inflatable Structures. Technical report, Jet Propulsion Laboratory, California Institute of Technology, October 1999. `http://trs-new.jpl.nasa.gov/dspace/bitstream/$2014/18370/1/99-1847$.pdf`.

[76] M. Thomas. Inflatable Space Structures. IEEE Potentials, December 1992.

[77] C.H. Jenkins and V.D. Kalanovic. Issues in Control of Space Membrane/Inflatable Structures. In *IEEE Aerospace Conference Proceedings*, volume 7, pages 411–414, 2000.

[78] M. Quadrelli. Modeling and Dynamics of Tethered Formations for Space Interferometry. In *The AAS/AIAA Spaceflight Mechanics Meeting*, February 2001.

[79] V. Szebelehely. *Theory of Orbits: The Restricted Three-body Problem*. Academic Press, 1967.

[80] O. Absil. Nulling Interferometry with IRSI-Darwin: Further Study of the Aperture Configurations. In J. Surdej, J.P. Swings, D. Caro, and A. Detal, editors, *Liege International Astrophysical Colloquia*, volume 36 of *Liege International Astrophysical Colloquia*, pages 79–84, 2001.

[81] O. Wallner, K. Ergenzinger, R. Flatscher, and U. Johann. DARWIN Mission and Configuration Trade-Off. In *Advances in Stellar Interferometry (Orlando), Proc. SPIE* 6268, Orlando, Florida, 2006.

[82] O. Wallner, K. Ergenzinger, R. Flatscher, and U. Johann. DARWIN Mission and Configuration Trade-Off . In *In the Proceedings of The International Society for Optical Engineering*, volume 6268, June 2006.

[83] The Terrestrial Planet Finder. `http://planetquest.jpl.nasa.gov`.

[84] R.W. Farquhar. The Control And Use of Liberation Point Satellites. Technical report, NASA, Washington D.C., September 1970. NASA Technical Report, TR R-346.

[85] S.M. Veres, S.B. Gabriel, D.Q Mayne, and E. Rogers. Analysis of Formation Flying Control of a Pair of Nanosatellites. *AIAA Journal of Guidance, Control and Dynamics*, 25(5):971–974, Sept-Oct 2002.

[86] S.M. Veres, D.Rokitianski, S.B. Gabriel, D.Q Mayne, and E. Rogers. Ellipsoid Methods For Formation Flying Control Of Two Nano-Satellites. In *Proceedings of the International Conference on Control Applications*, pages 298–303, Piscataway, USA, 2002.

[87] R. Pongvthithum, S.M. Veres, S.B. Gabriel, and E.Rogers. Universal Adaptive Control Of Satellite Formation Flying. *International Journal of Control*, 78(1):45–52, January 2005.

[88] D.Ya. Rokityanskiy and S.M. Veres. Application of Ellipsoidal Estimation to Satellite Control Design . *Mathematical and Computer Modelling of Dynamical Systems*, 11(2):239–249, June 2005.

[89] K.K.T Thanapalan, S.M. Veres, E. Rogers, and S.B. Gabriel. Fault Tolerant Controller Design to Ensure Operational Safety in Satellite Formation Flying. In *Proceedings of the* 45*th IEEE Conference on Decision and Control*, pages 1562–1567, San Diego, USA, December 2006.

[90] S.M. Veres. Autonomous Formation Flying Of Satellite Robots: The Mechanical Control Layer. In *Towards Autonomous Robotic Systems: TAROS*, Guildford, UK, September 2006.

[91] S.M Veres, K. Thanapalan, S.B. Gabriel, and E. Rogers. Reconfigurable Controller Design For Operational Safety In Satellite Formation Flying. In *International Control Conference* 2006, Glasgow, Scotland, UK, 30 Aug - 1 Sept 2006.

[92] S.M. Veres. Editorial (to the special issue on autonomous adaptive control of vehicle systems). *International Journal of Adaptive Control and Signal Processing*, 21(1), February 2007.

[93] F.Y. Hsiao and D.J. Scheeres. The Dynamics of Formation Flight about a Stable Trajectory. *The Journal of the Astronautical Sciences*, 50(3):269–287, 2002.

[94] K.C. Howell and B.G. Marchand. Natural and Non-natural Spacecraft Formations Near the $L1$ and $L2$ Liberation Points in The Sun-Earth/Moon Ephemeris System. *Dynamical Systems: An International Journal*, 20(1):149–173, 2005.

[95] G. Gómez, K. Howell, J. Masdemont, and C. Simó. Station-Keeping Strategies for Translunar Libration Point Orbits. In *AAS/AIAA Space Flight Mechanics Conference*, pages 98–168, Monterey, California, February 1998.

[96] K. Howell and L. Millard. Control of Satellite Imaging Formations in Multi-Body Regimes. In *IAF* 57*th International Astronautical Congress*, Valencia, Spain, October 2006.

[97] B.G. Marchand and K.C.Howell. Control Strategies for Formation Flight in the Vicinity of the Liberation Points. *Journal of Guidance Control and Dynamics*, 28(6):1210–1219, 2005.

[98] O. Junge, J.E. Marsden, and S. Ober-Blöbaum. Optimal reconfiguration of Formation Flying Spacecraft- a Decentralized Approach. In *Proceedings of the* 45*th IEEE Conference on Decision & Control*, San Diego, CA, USA, December 2006.

[99] B. Wie. *Space Vehicle Dynamics And Control*. AIAA Education Series, 1998.

[100] J.E. Kulkarni, M.E. Campbell, and G.E. Dullerud. Stabilization of Spacecraft Flight in Halo Orbits: An H∞ approach. *IEEE Transactions on Control Systems Technology*, 14(3):572–578, May 2006.

[101] H. Wong and V. Kapila. Spacecraft Formation Flying near Sun-Earth $L2$ Lagrange Point: Trajectory Generation and Adaptive Output Feedback Control. In 2005 *American Control Conference*, Portland, OR, USA, June 2005.

[102] V. Gazi. Swarm Aggregations Using Artificial Potentials and Sliding Mode Control. In *Proceedings of the IEEE Confereence on Descision and Control*, pages 2848–2853, December 2003.

[103] C.M. Saaj, V. Lappas, and V. Gazi. Spacecraft Swarm Navigation and Control Using Artificial Potential Field and Sliding Mode Control. In *IEEE International Conference on Industrial Technology*, pages 2646–2651, December 2006.

[104] M. Beckman. Orbit Determination Issues for Libration Point Orbits. In *International Conference of Libration Points Orbits and Applications*, Girona, Spain, June 2002.

[105] D. Folta, C. Gramling, A. Long, D. Leung, and S. Belur. Autonomous Navigation Using Celestial Objects. In K.C. Howell, F.R. Hoots, B. Kaufman, and K.T. Alfriend, editors, *Advances in the Astronautical Sciences, Astrodynamics* 1999, pages 2161–2179, 1999.

[106] S.M. Veres. Autonomous Control Systems Using Agents: An Introduction. In *Proceedings of IEE Workshop on Agent Based Control Systems*, pages 1–10, 2005.

[107] S.M. Veres and A.G. Veres. Learning and Adaptation in Physical Agents. In 9*th IFAC Workshop Adaptation and Learning in Control and Signal Processing*, St Petersburg, Russia, August 2007.

[108] S.M. Veres and A.G. Veres. Learning and Adaptation of Skills in Autonomous Physical Agents. In 17*th World Congress of International Federation of Automatic Control (IFAC)*, Seoul, Korea, July 2008.

[109] M.J. Mataric. Designing emergent behaviors: From local interactions to collective intelligence. In *Proceedings of The Second International Conference on From Animals To Animats* 2*: Simulation of Adaptive Behavior*, pages 432–441, Honolulu, Hawai, United States, 1993. MIT Press.

[110] A.F.T Winfield, C.J. Harper, and J. Nembrini. Towards Dependable Swarms and a New Discipline of Swarm Engineering. In *Simulation of Adaptive Behaviour, workshop on Swarm Robotics SAB'*04, pages 126–142, Santa Monica, CA, USA, 2005.

[111] T. Balch and R.C. Arkin. Behavior-Based Formation Control For Multirobot Teams. *Robotics and Automation, IEEE Transactions on*, 14(6):926–939, 1998.

[112] M.B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-Based Multirobot Coordination: A Survey and Analysis. In *In Proceedings of the IEEE*, pages 1257–1270, July 2006.

[113] C. Tovey, M.G. Lagoudakis, S. Jain, and S. Koenig. The Generation of Bidding Rules For Auction-Based Robot Coordination. In L.E. Parker, F.E. Schneider, and A.C. Schultz, editors, *Multi-Robot Systems: From Swarms to Intelligent Automata Volume III. Proceedings from the* 2005 *International Workshop on Multi-Robot Systems*, 2005.

[114] M.G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and S. Jain. Auction-Based Multi-Robot Routing. In *In Robotics: Science and Systems*, pages 343–350, 2005.

[115] V. Frias-martinez and S. Parsons. Exploring Auction Mechanisms For Role Assignment In Teams of Autonomous Robots. In *In Proceedings of the RoboCup Symposium*, pages 758–786, 2004.

[116] D.L. Richardson. Analytic Construction of Periodic Orbits About The Collinear Points. *Celestial Mechanics*, 22(1):241–253, February 1980.

[117] K.C. Howell. Three Dimensional Periodic 'Halo' Orbits. *Celestial Mechanics*, 32(1):53–71, 1984.

[118] A. McInnes. An Introduction To Liberation Point Orbits. Technical report, University of Canterbury, NZ, January 2006.

[119] The Solar and Heliospheric Observatory (SOHO). `http://sohowww.nascom.nasa.gov`.

[120] R. Thurman and P.A. Worfolk. The Geometry of Halo Orbits in The Circular Restricted Three-Body Problem. Technical report, The Geometry Center, University of Minnesota, 1996. Technical Report GCG95.

[121] J.V. Breakwell and J.V. Brown. The 'Halo' Family of 3-Dimensional Periodic Orbits in the Earth-Moon Restricted 3-Body Problem. *Celestial Mechanics*, 20(1):389–404, 1979.

[122] J. Senent, C. Ocampo, and A. Capella. Low-Thrust Variable Specific Impulse Transfers and Guidance to Unstable Periodic Orbits. *Journal of Guidance and Control*, 28(2):280–290, 2005.

[123] M.J. Sidi. *Spacecraft Dynamics And Control: A Practical Engineering Approach*. Cambridge University Press, 1997.

[124] J.L Meriam and L.G. Kraige. *Engineering Mechanics: Dynamics*. John Wiley and Sons LTD, 1993.

[125] J.B. Kuipers. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality*. Princeton University Press, 2002.

[126] R. Kristiansen. *Dynamic Synchronization of Spacecraft Modeling and Coordinated Control of Leader-Follower Spacecraft Formations*. PhD thesis, Norwegian University of Science and Technology, 2008.

[127] J. Roberts. Development of a Relative Motion Model for Satellite Formation Flying Around *L*2. Technical report, Space Research Center, Cranfield University, 2004.

[128] N.A. Shneydor. *Missile Guidance and Pursuit: Kinematics, Dynamics and Control*. Horwood Publishing Ltd, 1998.

[129] M. Breivik and T.I. Fossen. Guidance Laws for Planar Motion Control. In *Proceedings of the* 47^{th} *IEEE Conference on Desicion and Control*, pages 570–577, 2008.

[130] C.L. Lin and H.W. Su. Intelligent Control Theory in Guidance and Control System Design. In *Proceedings of the National Science Council, ROC*, pages 15–30, 2000.

[131] H.L. Pastrick, S.M. Seltzer, and M.E. Warren. Guidance laws for short-range tactical missiles. *Journal of Guidance Control and Dynamics*, 4(2):98–108, 1981.

[132] R.H. Battin. Space Guidance Evolution-A Personal Narrative. *Journal of Guidance Control and Dynamics*, 5(2):97–110, 1982.

[133] D. Izzo and L. Pettazzi. Autonomous and Distributed Motion Planning for Satellite Swarm. *Journal of Guidance Control and Dynamics*, 30(2):449–459, March-April 2007.

[134] K.Fletcher, M.Ayre, and N.Lan. *Biomimetic Engineering for Space Applications*. ESA Publications Division, 2006.

[135] O. Faugeras. *Three-dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1994.

[136] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[137] M.S.Grewal and A.P. Andrews. *Kalman Filtering: Theory and Proctice using MATLAB*. John Wiley and Sons LTD, 2001.

[138] D. Simon. *Optimal State Estimation: Kalman, H-inifinty and Nonlinear Approaches*. John Wiley and Sons LTD, 2006.

[139] O'Navi Inertial Sensor Technology website. `http://www.o-navi.com`.

[140] P. David, D. DeMenthon, R. Duraiswami, and H. Samet. SoftPOSIT: Simultaneous Pose and Correspondence Determination. *International Journal of Computer Vision*, 59(3):259–284, 2004.

[141] E.J. Lefferts, F.L.Markley, and M.D. Schuster. Kalman Filtering for Spacecraft Attitude Estimation. *Journal of Guidance Control and Dynamics*, 5(5):417–429, Sept-Oct 1982.

[142] G. Creamer. Spacecraft Attitude Determination Using Gyros and Quaternion Measurements. *Journal of the Astronautical Sciences*, 44(3):357–371, 1996.

[143] G. Creamer. Spacecraft Attitude Determination Using Gyros and Wide Field-Of-View Cameras. In *Proceedings of the Guidance Navigation and Control Conference*, San Diego, CA, USA, 1996.

[144] A. Wu and D.H. Hein. Stellar inertial attitude determination for leo spacecraft. In *Proceedings of the* 35^{th} *IEEE Conference on Desicion and Control*, pages 3236–3244, Kobe, Japan, 1996.

[145] K.J.Astrom and B. Wittenmark. *Computer Controlled Systems Theroy and Design*. Prentice Hall Information and System Sciences Series, 1997.

[146] P.S.Maybeck. *Stochastic Models, Estimation and Control: Volume* 1. New York Academic Press, 1997.

[147] N.Trawny and S.I. Roumeliotis. Indirect Kalman Filter for 3D Attitude Estimation: A Tutorial For Quaternion Algebra. Technical report, Multiple Autonomous Robotic Systems Laboratory, March 2005. Technical report number $2005-002$, revision 57.

[148] N.K. Lincoln and S.M. Veres. Components of a Vision Assisted Constrained Autonomous Satellite Formation Flying Control System. *International Journal of Adaptive Control and Signal Processing*, 21(2-3):237–264, October 2006.

[149] V.I. Utkin, J. Guldner, and J. Shi. *Sliding Mode Control in Electromechanical Systems*. CRC Press, 1999.

[150] G. Nikolski. On Automatic Stability of a Ship on a Given Course (in Russian). In *Proceedings of the Central Communications Laboratory*, pages 34–75, 1934.

[151] W. Gao and J.C. Hung. Variable Structure Control: A Survey. *IEEE Transactions on Industrial Electronics*, 40(1):2–22, February 1993.

[152] R. DeCarlo, S.H. Zak, and G.P. Matthews. Variable Structure Control of Nonlinear Multivariable Systems: A Tutorial. In *Proceedings of the IEEE*, pages 212–232, March 1988.

[153] V. Utkin. Adaptive Discrete Time Sliding Mode Control of Infinite Dimensional Systems. In *Proceedings of the* 37*th IEEE Conference on Decision and Control*, pages 4033–4038, Tempa, Florida, USA, December 1998.

[154] A.J. Koshkouei. Sliding Mode Control of Discrete-Time Systems. *Journal of Dynamic Systems, Measurement and Control*, 122:793–801, December 2000.

[155] W. Gao, Y. Wang, and A. Homaifa. Discrete-Time Variable Structure Control Systems. *IEE Transactions on Industrial Electronics*, 42(2):117–122, April 1995.

[156] B. Bandyopadhyay and V.K. Thakar. Discrete Time Output Feedback Sliding Mode Control For Nonlinear MIMO System: A Stepper Motor Case. *International Journal of Systems Science*, 39(1):89–104, January 2008.

[157] T. Schouwenaars, B. De-Moor, E. Feron, and J. How. Mixed Integer Programming For Multi-Vehicle Path Planning. In *Proceedings of the European Control Conference* 2001, pages 2603–2608, 2001.

[158] A. Richards and J.P. How. Aircraft Trajectory Planning with Collision Avoidance using Mixed Integer Linear Programming. In *Proceedings of the American Control Conference* 2002, pages 1936–1941, 2002.

[159] GLPK. `http://www.kivasystems.com`.

[160] ILOG CPLEX. `http://www.ilog.com`.

[161] R.Fourer, D.M.Gay, and B.W. Kernighan. *AMPL:A Modelling Language For Mathematical Programming*. Curt Hinrichs, 2003.

[162] AMPL. `http://www.ampl.com`.

[163] N.K. Lincoln and S.M. Veres. Six Degree of Freedom Variable Hierarchy Sliding Mode Control in Halo OrbitsWith Potential Function Guidance. In *Proceedings of the* 47^{th} *IEEE Conference on Decision and Control*, Cancun, Mexico, 2008.

[164] A. Frank. On Kuhns Hungarian Method A Tribute From Hungary. Technical report, Egervary Research Group, Budapest, Hungary, October 2004. Egervary Research Group on Combinatorial Optimization.

[165] A. Sahu and R. Tapadar. Solving the Assignment Problem Using Genetic Algorithm and Simulated Annealing. *International Journal of Applied Mathematics*, 36(1), February 2007.

[166] Online NEOS server. `http://www-neos.mcs.anl.gov`.

[167] Tomlab Website. `http://tomopt.com`.

[168] L.A. Dennis. Agent Infrastructure Layer (AIL): Design and Operational Semantics. Technical report, Department of Computer Science, University of Liverpool, 2007.

[169] L.A. Dennis, B. Farwer, R.H. Bordini, M. Fisher, and M. Wooldridge. A Common Semantic Basis for BDI Languages. In *Proceedings of the* 7^{th} *International Workshop on Programming Multiagent Systems (ProMAS)*, pages 88–103, Honolulu, Hawaii, USA, 2007.

[170] S.M. Veres and J.Luo. Formal Verification of Autonomous Control Agents. In *Proceedings of the* 6*th IASTED International Conference: Intelligent Systems and Control*, pages 140–144, Honolulu, Hawaii, USA, August 2004.

[171] J. Lygeros, C. Tomlin, and S. Sastry. Controllers For Reachability Specifications For Hybrid Systems. *Automatica*, 35:349–370, 1999.

[172] sEnglish Website. `http://system-english.com`.

[173] S.M. Veres and N.K. Lincoln. Sliding Mode Control for Agents and Humans-the use of sEnglish for publications. In *Proceedings of TAROS* 2008*, Towards Autonomous Robotic Systems*, Edinburgh, Scotland, 2008.

[174] The OpenCV (Open Comuter Vision). `http://opencv.willowgarage.com/wiki`.

[175] Foundation for Intelligent Physical Agents (FIPA). `http://www.fipa.org`.

[176] J.M. Hendrix, B.D.O. Anderson, and V.D.Blondel. Rigitity and Persistence of Directed Graphs. In *Proceedings of the* 44^{th} *Conference on Decision and Control*, Seville, Spain, 2005.

Part IV

APPENDICES

Appendix A

INTERFEROMETER APERTURE POINT LOCATIONS

A.1 Golay Arrays

Table A.1 provides the threefold symmetric Golay array subaperture coordinates, in non-dimensional form, as extracted from [69]. Table A.1 lists the orthogonal Golay array subaperture points, in non-dimensional form, as inferred from [71] and [61].

Golay-3	$\left(-\frac{1}{2},-\frac{\sqrt{3}}{6}\right)$	$\left(\frac{1}{2},-\frac{\sqrt{3}}{6}\right)$	$\left(0,\frac{\sqrt{3}}{3}\right)$
Golay-6	$\left(0,-\frac{2\sqrt{3}}{3}\right)$ $\left(\frac{1}{2},\frac{5\sqrt{3}}{6}\right)$	$\left(1,-\frac{2\sqrt{3}}{3}\right)$ $\left(-1,-\frac{\sqrt{3}}{3}\right)$	$\left(1,\frac{\sqrt{3}}{3}\right)$ $\left(-\frac{3}{2},-\frac{\sqrt{3}}{6}\right)$
Golay-9	$\left(-\frac{1}{2},-\frac{\sqrt{3}}{6}\right)$ $\left(\frac{7}{2},-\frac{\sqrt{3}}{6}\right)$ $\left(-2,\frac{\sqrt{3}}{3}\right)$	$\left(\frac{1}{2},-\frac{\sqrt{3}}{6}\right)$ $\left(\frac{3}{2},\frac{5\sqrt{3}}{6}\right)$ $\left(-2,-\frac{5\sqrt{3}}{3}\right)$	$\left(0,\frac{\sqrt{3}}{3}\right)$ $\left(-\frac{3}{2},\frac{11\sqrt{3}}{6}\right)$ $\left(\frac{1}{2},-\frac{7\sqrt{3}}{6}\right)$
Golay-12	$\left(0,-\sqrt{3}\right)$ $\left(-4,-2\sqrt{3}\right)$ $\left(\frac{5}{2},\frac{\sqrt{3}}{2}\right)$ $\left(-\frac{3}{2},\frac{\sqrt{3}}{2}\right)$	$\left(-2,-\sqrt{3}\right)$ $\left(5,-\sqrt{3}\right)$ $\left(\frac{3}{2},\frac{\sqrt{3}}{2}\right)$ $\left(-2,\sqrt{3}\right)$	$\left(-\frac{1}{2},-\frac{3\sqrt{3}}{2}\right)$ $\left(\frac{5}{2},-\frac{\sqrt{3}}{2}\right)$ $\left(-\frac{1}{2},\frac{3\sqrt{3}}{2}\right)$ $\left(-1,3\sqrt{3}\right)$

Tab. A.1: Table of non-dimensional threefold symmetric Golay array subaperature points.

Golay-3	$(0,0)$	$(1,0)$	$(0,1)$
Golay-4	$(-2,-3)$ $(0,-2)$	$(-1,-1)$	$(-1,-2)$
Golay-5	$(-2,-3)$ $(0,-2)$	$(-3,-1)$ $(-1,-1)$	$(-1,-2)$
Golay-6	$(-2,-3)$ $(0,-2)$	$(-3,-1)$ $(-1,1)$	$(-1,-2)$ $(-1,-1)$
Golay-7	$(-2,-3)$ $(2,-4)$ $(0,-2)$	$(-3,-1)$ $(-1,1)$	$(-1,-2)$ $(-1,-1)$
Golay-8	$(2,1)$ $(2,-4)$ $(0,-2)$	$(-3,-1)$ $(-1,1)$ $(-2,-3)$	$(-1,-2)$ $(-1,-1)$
Golay-9	$(2,1)$ $(2,-4)$ $(0,-2)$	$(-3,-1)$ $(0,4)$ $(-2,-3)$	$(-1,-2)$ $(-1,-1)$ $(-1,1)$
Golay-10	$(2,1)$ $(2,-4)$ $(0,-2)$ $(-2,-3)$	$(-3,-1)$ $(0,4)$ $(5,-3)$	$(-1,-2)$ $(-1,-1)$ $(-1,1)$

Tab. A.2: Table of non-dimensional orthogonal Golay array subaperature points.

Appendix B

SUPPLEMENTAL MATERIAL RELATING TO THE THREE BODY PROBLEM

B.1 Second Partial Derivatives of the Three-Body Problem

The following are generalized expressions for the second partial derivatives of U, the pseudo-potential for the CRTBP, used within (6.11) of Chapter 6. Each derivative is denoted as $U_{jk} = \frac{\partial U}{\partial_j \partial_k}$, where $j,k \in (x,y,z)$.

$$
\begin{aligned}
U_{xx} &= 1 - \frac{(1-\mu)}{r_1^3} - \frac{\mu}{r_2^3} + \frac{3(1-\mu)(x+\mu)^2}{r_1^5} + \frac{3\mu(x-(1-\mu))^2}{r_2^5}, \\
U_{xy} &= \frac{3(1-\mu)(x+\mu)y}{r_1^5} + \frac{3\mu(x-(1-\mu))y}{r_2^5}, \\
U_{xz} &= \frac{3(1-\mu)(x+\mu)z}{r_1^5} + \frac{3\mu(x-(1-\mu))z}{r_2^5}, \\
U_{yx} &= \frac{3(1-\mu)(x+\mu)y}{r_1^5} + \frac{3\mu(x-(1-\mu))y}{r_2^5}, \\
U_{yy} &= 1 - \frac{(1-\mu)}{r_1^3} - \frac{\mu}{r_2^3} + \frac{3(1-\mu)y^2}{r_1^5} + \frac{3\mu y^2}{r_2^5}, \\
U_{yz} &= \frac{3(1-\mu)yz}{r_1^5} + \frac{3\mu yz}{r_2^5}, \\
U_{zx} &= \frac{3(1-\mu)(x+\mu)z}{r_1^5} + \frac{3\mu(x-(1-\mu))z}{r_2^5}, \\
U_{zy} &= \frac{3(1-\mu)yz}{r_1^5} + \frac{3\mu yz}{r_2^5}, \\
U_{zz} &= -\frac{(1-\mu)}{r_1^3} - \frac{\mu}{r_2^3} + \frac{3(1-\mu)z^2}{r_1^5} + \frac{3\mu z^2}{r_2^5}.
\end{aligned}
$$

where

$$\begin{aligned} \mu &= \frac{M_2}{M_1+M_2} \\ &= (1-\rho) \end{aligned}$$

$$\begin{aligned} r_1 &= \sqrt{(X_0-\rho)^2-Y_0^2} \\ r_2 &= \sqrt{(X_0+1-\rho)^2+Y_0^2} \end{aligned}$$

B.2 Constants For Richardson Approximation

This section details all the constants required for use within the Richardson approximation to an analytical halo orbit about one of the collinear liberation points presented within Chapter 6. The details of all the below equations were taken from [120] and contained within this thesis for completion only.

Note that whenever the double sign appears, the upper sign applies to the L_1 point and the lower to the L_2 point. D corresponds to the primary mass inter-distance and r_i is the distance between liberation point i and the closest primary.

For L_1 or L_2:

$$\begin{aligned} \delta &= -\frac{r_i}{D} \\ c_n &= (\pm 1)^n \frac{\mu}{|\delta-(1-\mu)|^3} + (-1)^n(1-\mu)\frac{|\delta-(1-\mu)|^{n-2}}{|\delta+\mu|^{n+1}} \end{aligned}$$

For L_3:

$$c_n = (-1)^n \frac{1-\mu}{|r_2|^3} + (-1)^n \frac{\mu |r_2|^{n-2}}{|r_1|^{n+1}}$$

$$
\begin{aligned}
\zeta &= (-1)^n \\
\lambda &= \sqrt{\frac{1}{2}(2-c_2+\sqrt{9c_2^2-8c_2})} \\
K &= \frac{2\lambda}{\lambda^2+1-c_2} \\
\alpha_1 &= -\frac{3}{4}c_3[A_x^2(K^2-2)+A_z^2] \\
D_1 &= 16\lambda^4+4\lambda 2(c_2-2)-2c_2^2+c_2+1 \\
D_2 &= 81\lambda^4+9\lambda^2(c_2-2)-2c_2^2+c_2+1 \\
D_3 &= 2\lambda[\lambda(1+K^2)-2K] \\
s_1 &= \frac{1}{D_3}\left\{\frac{3}{2}c_3[2a_{21}(K^2-2)-a_{23}(K^2+2)-2Kb_{21}]-\frac{3}{8}c_4[3k^4-8K^2+8]\right\} \\
s_2 &= \frac{1}{D_3}\left\{\frac{3}{2}c_3[2a_{22}(K^2-2)-\zeta_{24}(K^2+2)-2\zeta Kb_{22}+d_{21}(2-3\zeta)]+\frac{3}{8}c_4[(8-4\zeta)-K^2(2+\zeta)]\right\}
\end{aligned}
$$

$$
\begin{aligned}
a_{21} &= \frac{3c_3(k^2-2)}{4(1+2c_2)} \\
a_{22} &= \frac{3c_3}{4(1+2c_2)} \\
a_{23} &= -\frac{3\lambda c_3}{4KD_1}(3K^3\lambda-6K(K-\lambda)+4) \\
a_{24} &= -\frac{3\lambda c_3}{4KD_1}(2+3K\lambda) \\
a_{31} &= -\frac{9\lambda}{D_2}[c_3(Ka_{23}-b_{21})+Kc_4(1+\frac{1}{4}K^2)]+\frac{9\lambda^2+1-c_2}{2D_2}[3c_3(2a_{23}-Kb_{21})+c_4(2+3K^2)] \\
a_{32} &= -\frac{9\lambda}{4D_2}[4c_3(Ka_{24}-b_{22})+Kc_4]-\frac{3(9\lambda^2+1-c_2}{2D_2}[c_3(Kb_{22}+d_{21}-2a_{24})-c_4]
\end{aligned}
$$

$$
\begin{aligned}
b_{21} &= -\frac{3c_3\lambda}{2D_1}(3\lambda K-4)\\
b_{22} &= \frac{3\lambda c_3}{D_1}\\
b_{31} &= \frac{1}{D_2}[3\lambda(3c_3(Kb_{21}-2a_{23})-c_4(2+3K^2))+\frac{1}{8}(9\lambda^2+1+2c_2)(12c_3(Ka_{23}-b_{21})+3Kc_4(4+K^2)]\\
b_{32} &= \frac{1}{D_2}[3\lambda(3c_3(kb_{22}+d_{21}-2a_{24})-3c_4)+\frac{1}{8}(9\lambda^2+1+2c_2)(12c_3(ka_{24}-b_{22}))]\\
b_{33} &= -\frac{K}{16\lambda}[12c_3(b_{21}-2Ka_{21}+Ka_{23})+3c_4K(3K^2-4)+16s_1\lambda(\lambda K-1)]\\
b_{34} &= -\frac{K}{8\lambda}[-12c_3Ka_{22}+3c_4K+8s_2\lambda(\lambda K-1)]\\
b_{35} &= -\frac{K}{16\lambda}[12c_3(b_{22}+Ka_{24})+3c_4K]\\
d_{21} &= -\frac{c_3}{2\lambda^2}\\
d_{31} &= \frac{3}{64\lambda^2}(4c_3a_{24}+c_4)\\
d_{32} &= \frac{3}{64\lambda^2}[4c_3(a_{23}-d_{21})+c_4(4+K^2)]
\end{aligned}
$$

Appendix C

AMPL FILES

This section contains the AMPL model and data files used within the optimization of agent positioning throughout the mission lifetime. If access to a full version of AMPL with CPLEX is available, the problem can be read in and solved within the AMPL shell environment using the following commands:

```
>>model AgentMotion.mod;
>>data AgentMotion.dat;
>>options solver cplex;
>>solve;
```

If access to a full version of AMPL with CPLEX is not available, it is suggested that a trial version of Tomlab be used to interface the AMPL files using MATLAB. Within the AMPL shell environment, the following commands are required:

```
>>model AgentMotion.mod;
>>data AgentMotion.dat;
>>write gAgentMotion;
```

This will generate a data construct file named `AgentMotion.nl` which is to be used by the Tomlab program in forming a solution. Ensure that the model, data and construct files are within the working directory of MATLAB and use the following commands:

```
>>Problem=amplAssign('AgentMotion',1,1);
>>Solution=tomRun('cplex',Problem,1);
```

The resultant solution file can then be interfaced using the AMPL shell environment using:

```
>>model AgentMotion.mod;
>>data AgentMotion.dat;
>>solution AgentMotion.sol;
```

Whereupon all AMPL commands for display and data format may be used.

C.1 AMPL Model File

```
#Model file for agent motion
#Maximize the resultant fuel reserves at the end of all completed 'orbits'
#
####################
#SET DATA:
   set Agents;
   set Positions;
####################
#PARAMETER DATA:
#
#Number of orbits to complete
    param n integer >0;
#initial agent fuel
    param InitialFuel{j in Agents};
#minimum fuel allowance
    param MinFuel{j in Agents};
#Maintenance cost for position per orbit
    param COST1{j in Agents,k in Positions}>0;
#Position swap costs
    param COST2{k in Positions,P in Positions}>=0;
#Initial Positions (inferred by null swap)
    param Swap0{j in Agents, k in Positions, P in Positions, t in 0..n}
        binary;
#
####################
#VARIABLES:
#
#Agent fuel status
    var AgentFuel{j in Agents, t in 0..(n+1)} >=0;
#Agent pos switching(s)
    var AgentSwap{j in Agents,k in Positions, P in Positions, t in 0..(n+1)}
        binary;
#Cost of motion at time t
    var TotalCost{j in Agents, t in 0..n};
#Agent positions at time t
    var AgentPos{j in Agents, k in Positions, t in 0..n};
#
####################
#TASK: maximze agent reserves at end of all 'orbits'
#
    maximize FinalFuel: sum{j in Agents}AgentFuel[j,(n+1)];
#
####################
#CONDITIONS:
#
#-Initial Fuel
subject to InitalFuel {j in Agents}:
    AgentFuel[j,0]=InitialFuel[j];

#-initial pos allocation (inferred through null swap)
subject to StartPos {j in Agents, k in Positions, P in Positions}:
AgentSwap[j,k,P,0]=Swap0[j,k,P,0];
#-Fuel depletion and limits
subject to FuelLimit{j in Agents, t in 0..(n+1)}:
AgentFuel[j,t]>=MinFuel[j];
```

```
subject to FuelDepletion {j in Agents, t in 0..n}:
AgentFuel[j,(t+1)]=AgentFuel[j,t]- TotalCost[j,t];
subject to MotionCost{j in Agents, t in 0..n}:
    TotalCost[j,t]=
        sum{k in Positions,P in Positions}COST2[k,P]*AgentSwap[j,k,P,t]
        +sum{k in Positions}COST1[j,k]*sum{P in Positions}AgentSwap[j,k,P,t];
#AgentSwap conditions:
# -Must go or stay somewhere
subject to Restrict1{j in Agents, t in 0..n}:
    sum{k in Positions, P in Positions}AgentSwap[j,k,P,t]=1;
# -Can only come from where they were
subject to Restrict2{j in Agents,k in Positions, t in 0..n}:
sum{P in Positions}AgentSwap[j,k,P,(t+1)]
        =sum{P in Positions}AgentSwap[j,P,k,t];
# -can't have more than one in the position
subject to Restrict3{k in Positions,t in 0..n}:
sum{j in Agents,P in Positions}AgentSwap[j,k,P,t]<=1;
#keep track of agent positions
subject to KeepTrack{j in Agents,k in Positions, t in 0..n}:
AgentPos[j,k,t]=sum{P in Positions}AgentSwap[j,k,P,t];
```

C.2 AMPL Data File (Small Scale Problem)

```
set Agents:= A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 ;
set Positions:=P0 P1 P2 P3 P4 P5 P6 P7 P8 P9 ;
#BASIC PARAMETERS
param n:=12;

#COMPLEX PARAMETERS
param:   InitialFuel MinFuel :=
A0     94.1614      1
A1   91.3578      1
A2 93.147      1
A3   85.2528      1
A4   96.8808      1
A5   96.8808      1
A6   97.4628      1
A7   92.3692      1
A8   94.9257      1
A9   96.1846      1;

#Costs associated with agents being stationed at a position
param COST1 (tr):
A0  A1     A2    A3    A4    A5    A6    A7    A8    A9  :=
P0  5.8386  5.8386  5.8386  5.8386  5.8386  5.8386  5.8386  5.8386  5.8386  5.8386
P1  8.6422  8.6422  8.6422  8.6422  8.6422  8.6422  8.6422  8.6422  8.6422  8.6422
P2  6.8530  6.8530  6.8530  6.8530  6.8530  6.8530  6.8530  6.8530  6.8530  6.8530
P3 14.7472 14.7472 14.7472 14.7472 14.7472 14.7472 14.7472 14.7472 14.7472 14.7472
P4  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192
P5  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192  3.1192
P6  2.5372  2.5372  2.5372  2.5372  2.5372  2.5372  2.5372  2.5372  2.5372  2.5372
P7  7.6308  7.6308  7.6308  7.6308  7.6308  7.6308  7.6308  7.6308  7.6308  7.6308
P8  5.0743  5.0743  5.0743  5.0743  5.0743  5.0743  5.0743  5.0743  5.0743  5.0743
P9  3.8154  3.8154  3.8154  3.8154  3.8154  3.8154  3.8154  3.8154  3.8154  3.8154;
```

```
#Costs associated with agents swapping position
param COST2 (tr):
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0 0     1.957 2.085 1.8603 1.1346 1.3742 1.3742 1.7969 1.3742 1.6045
P1 1.957 0     0.8701 2.7911 1.0965 0.7753 1.2013 2.125 2.125 0.8701
P2 2.085 0.8701 0     2.3987 1.5263 0.8701 0.8701 1.5263 2.5208 0.5586
P3 1.8603 2.7911 2.3987 0     2.5694 2.2426 1.8398 1.2013 2.9971 2.1411
P4 1.1346 1.0965 1.5263 2.5694 0     0.7753 1.2013 2.125 1.2013 1.1346
P5 1.3742 0.7753 0.8701 2.2426 0.7753 0     0.5586 1.6045 1.8398 0.395
P6 1.3742 1.2013 0.8701 1.8398 1.2013 0.5586 0     1.0965 2.1044 0.395
P7 1.7969 2.125 1.5263 1.2013 2.125 1.6045 1.0965 0     2.7911 1.3742
P8 1.3742 2.125 2.5208 2.9971 1.2013 1.8398 2.1044 2.7911 0     2.1411
P9 1.6045 0.8701 0.5586 2.1411 1.1346 0.395 0.395 1.3742 2.1411 0;

#initial position allocations for agents, inferred through null swap
param Swap0:=
[A0,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       1   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A1,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   1   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A2,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   1

[A3,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
```

```
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   1   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A4,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   1   0
P9       0   0   0  0  0   0   0   0   0   0

[A5,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   1   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A6,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   1   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A7,*,*,0]:
     P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  1  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
```

```
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A8,*,*,0]:
    P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  1   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A9,*,*,0]:
      P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   1  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0;
```

C.3 AMPL Data File (Large Scale Problem)

```
set Agents:= A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 ;
set Positions:=P0 P1 P2 P3 P4 P5 P6 P7 P8 P9 ;

#BASIC PARAMETERS
param n:=155;

#COMPLEX PARAMETERS

param:   InitialFuel MinFuel :=
A0   99.4161 0
A1   99.1358 0
A2   99.3147 0
A3   98.5253 0
A4   99.6881 0
A5   99.6881 0
A6   99.7463 0
A7   99.2369 0
A8   99.4926 0
A9   99.6185 0;

#Costs associated with agents being stationed at a position
```

```
param COST1 (tr):
A0  A1     A2    A3    A4    A5    A6    A7    A8    A9  :=
P0 0.58386  0.58386 0.58386 0.58386 0.58386 0.58386 0.58386 0.58386 0.58386 0.58386
P1 0.86422  0.86422 0.86422 0.86422 0.86422 0.86422 0.86422 0.86422 0.86422 0.86422
P2 0.68530  0.68530 0.68530 0.68530 0.68530 0.68530 0.68530 0.68530 0.68530 0.68530
P3 1.47472  1.47472 1.47472 1.47472 1.47472 1.47472 1.47472 1.47472 1.47472 1.47472
P4 0.31192  0.31192 0.31192 0.31192 0.31192 0.31192 0.31192 0.31192 0.31192 0.31192
P5 0.31192  0.31192 0.31192 0.31192 0.31192 0.31192 0.31192 0.31192 0.31192 0.31192
P6 0.25372  0.25372 0.25372 0.25372 0.25372 0.25372 0.25372 0.25372 0.25372 0.25372
P7 0.76308  0.76308 0.76308 0.76308 0.76308 0.76308 0.76308 0.76308 0.76308 0.76308
P8 0.50743  0.50743 0.50743 0.50743 0.50743 0.50743 0.50743 0.50743 0.50743 0.50743
P9 0.38154  0.38154 0.38154 0.38154 0.38154 0.38154 0.38154 0.38154 0.38154 0.38154;

#Costs associated with agents swapping position
param COST2 (tr):
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0      0    0.1957    0.2085    0.1860    0.1135    0.1374    0.1374    0.1797    0.1374    0.1605
P1 0.1957         0    0.0870    0.2791    0.1096    0.0775    0.1201    0.2125    0.2125    0.0870
P2 0.2085    0.0870         0    0.2399    0.1526    0.0870    0.0870    0.1526    0.2521    0.0559
P3 0.1860    0.2791    0.2399         0    0.2569    0.2243    0.1840    0.1201    0.2997    0.2141
P4 0.1135    0.1096    0.1526    0.2569         0    0.0775    0.1201    0.2125    0.1201    0.1135
P5 0.1374    0.0775    0.0870    0.2243    0.0775         0    0.0559    0.1605    0.1840    0.0395
P6 0.1374    0.1201    0.0870    0.1840    0.1201    0.0559         0    0.1096    0.2104    0.0395
P7 0.1797    0.2125    0.1526    0.1201    0.2125    0.1605    0.1096         0    0.2791    0.1374
P8 0.1374    0.2125    0.2521    0.2997    0.1201    0.1840    0.2104    0.2791         0    0.2141
P9 0.1605    0.0870    0.0559    0.2141    0.1135    0.0395    0.0395    0.1374    0.2141         0;

#initial position allocations for agents, inferred through null swap
param Swap0:=
[A0,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       1   0   0  0  0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0

[A1,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0
P1       0   1   0  0  0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0

[A2,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
```

```
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   1

[A3,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   1   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A4,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   1   0
P9       0   0   0  0  0   0   0   0   0   0

[A5,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   1   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A6,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
```

```
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   1   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A7,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  1  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A8,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   0  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  1   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0

[A9,*,*,0]:
P0  P1  P2  P3  P4 P5 P6  P7  P8  P9 :=
P0       0   0   0  0  0   0   0   0   0   0
P1       0   0   0  0  0   0   0   0   0   0
P2       0   0   1  0  0   0   0   0   0   0
P3       0   0   0  0  0   0   0   0   0   0
P4       0   0   0  0  0   0   0   0   0   0
P5       0   0   0  0  0   0   0   0   0   0
P6       0   0   0  0  0   0   0   0   0   0
P7       0   0   0  0  0   0   0   0   0   0
P8       0   0   0  0  0   0   0   0   0   0
P9       0   0   0  0  0   0   0   0   0   0;
```

Appendix D

SENGLISH PAPER ON CONTINUOUS TIME SLIDING MODE CONTROL

Continuous Time Sliding Mode Control For Spacecraft Agents

N.K.Lincoln
School of Engineering Sciences
University of Southampton, Email: nkl@soton.ac.uk

---

sEnglish for Scientist, Engineers and Agents
This document is understandable by any agent produced by the Cognitive Agents Toolbox (www.cognitive-agents-toolbox.com) and by any sEnglish Agent (www.system-english.com).

C O N T E N T S

This paper is a PDF version of a document that can be read by agents that have the ability to interpret sEnglish sentences. Paragraphs in italics, such as this, are informal English inserted by the human author. All non-italics text, including titles of sections are interpreted by agents.

D.1 Conceptual structures used in this paper

This section first outlines the main concepts and objects used, then the basic data object constraints and finally the attributes of concepts and objects that make up the substance of their meaning.

D.1.1 The main concept and their relationships

The main concepts are : attitude error, desired attitude, desired position, desired state, position error, spacecraft forces, spacecraft movements . These do not have any sup classes, they represent root concepts.

The concepts that are subclasses of larger complex classes

An global memory is a special case of cell array and vector.

The special cases of the most significant concept classes

Special cases of cell array are global memory and memory item.

Special cases of state are position, omega, quaternion, quaternion error and velocity.

Attributes of an actuator limits

An 'actuator limits' has the following properties: its 'upper bounds' that is a number array and its 'lower bounds' that is a number array.

Attributes of an angular velocity

An 'angular velocity' has the following properties: its 'dimension' that can be one of a {'rad/s'}.

Attributes of an angular velocity error

An 'angular velocity error' has the following properties: its 'dimension' that can be one of a {'rad/s'}.

Attributes of an attitude error

An 'attitude error' has the following properties: its 'dimension' that can be one of a {'euler angles', 'quaternion'}, its 'values' that is a vector and its 'reference frame' that is a text.

Attributes of a cell array

A 'cell array' has the following properties:

Attributes of a control force

A 'control force' has the following properties: its 'dimension' that can be one of a {'nm','mn'}.

Attributes of a control torque

A 'control torque' has the following properties: its 'dimension' that can be one of a {'nm','mnm'}.

Attributes of a current state

A 'current state' has the following properties:

Attributes of a desired attitude

A 'desired attitude' has the following properties: its 'dimension' that can be one of a {'euler angles', 'quaternion'}, its 'values' that is a vector and its 'reference frame' that is a text.

Attributes of a desired position

A 'desired position' has the following properties: its 'dimension' that can be one of a {'km','m','mm'}, its 'values' that is a vector and its 'reference frame' that is a text.

Attributes of a desired state

A 'desired state' has the following properties: its 'dimensions' that is a set of char, its 'values' that is a vector, its 'reference frame' that is a text and its 'time horizon' that is a number array.

Attributes of a dynamical force

A 'dynamical force' has the following properties: its 'dimension' that can be one of a {'n','mn'}.

Attributes of a global memory

A 'global memory' has the following properties:

Attributes of a guidance direction

A 'guidance direction' has the following properties:

Attributes of a guidance kinematics

A 'guidance kinematics' has the following properties:

Attributes of a guidance omega gradient

A 'guidance omega gradient' has the following properties:

Attributes of a guidance reference

A 'guidance reference' has the following properties:

Attributes of an inertia matrix

An 'inertia matrix' has the following properties: its 'dimension' that can be one of a {'kgm^{-2}'}.

Attributes of a joint sliding surface

A 'joint sliding surface' has the following properties:

Attributes of a position error

A 'position error' has the following properties: its 'dimension' that can be one of a {'km','m','mm'}, its 'values' that is a vector and its 'reference frame' that is a text.

Attributes of a potential gains

A 'potential gains' has the following properties:

Attributes of an smoothed sign function

An 'smoothed sign function' has the following properties:

Attributes of an spacecraft forces

An 'spacecraft forces' has the following properties: its 'time axis' that is a number array and its 'forces' that is a set of vectors.

Attributes of an spacecraft movements

An 'spacecraft movements' has the following properties: its 'time axis' that is a number array, its 'rotations' that is a set of quaternions and its 'translations' that is a set of vectors.

Attributes of an state

An 'state' has the following properties:

Attributes of an state derivative

An 'state derivative' has the following properties:

Attributes of an state error

An 'state error' has the following properties:

Attributes of a surface weights

A 'surface weights' has the following properties:

Attributes of a translational velocity

A 'translational velocity' has the following properties: its 'dimension' that can be one of a {'km/s','m/s'}.

Attributes of a translational velocity error

A 'translational velocity error' has the following properties: its 'dimension' that can be one of a {'km/s','m/s'}.

D.2 Main Usage

D.2.1 Determine control response

Sentences to use:

Determine control torque Tout and control force Fout based upon desired state Xdes and global memory M .

Available things are: Xdes(desired state) , M(global memory) .

Details of the meaning:

Obtain current state Xnow from sensors observing state X . Determine state error Xe between state Xnow and desired state Xdes . Retrieve 'PotentialGains' memory item Pg from global memory M . Determine guidance kinematics G by potential functions using state error Xe and potential gains Pg . Retrieve 'SurfaceWeights' memory item Sw from global memory M . Build joint sliding surface S using current state Xnow , surface weights Sw and guidance kinematics G . Retrieve 'SurfaceLimit' memory item Sl from global memory M . Form modified joint sliding surface S2 by saturatating joint sliding surface S according to surface limit Sl . Retrieve 'Mass' memory item SAm from global memory M . Retrieve 'InertiaMatrix' memory item SAj from global memory M . Determine control torque T and control force F using joint sliding surface S2 , surface weights Sw , spacecraft agent mass SAm , spacecraft agent inertia matrix SAj and current state Xnow . Retrieve 'ActuatorLimits' memory item Ubounds from global memory M . Formulate output control torque Tout and control force Fout by saturating control torque T and control force F with actuator limits Ubounds .

Resulting things are: Tout(control torque) , Fout(control force) .

D.3 Component Sentances

D.3.1 Form joint sliding surfaces

Sentences to use:

Build joint sliding surface S using current state Xnow , surface weights Sw and guidance kinematics G .

Available things are: Xnow(current state) , Sw(surface weights) , G(guidance kinematics) .

Details of the meaning:

Resulting things are: S(joint sliding surface) .

D.3.2 Determine unbounded control

Sentences to use:

Determine control torque T and control force F using joint sliding surface S2 , surface weights Sw , spacecraft agent mass SAm , spacecraft agent inertia matrix SAj and current state Xnow .

Available things are: S2(joint sliding surface) , SAm(spacecraft agent mass) , SAj(inertia matrix) , Xnow(current state) , Sw(surface weights) .

Details of the meaning:

Resulting things are: T(control torque) , F(control force) .

D.3.3 Potential function guidance

Sentences to use:

Determine guidance kinematics G by potential functions using state error Xe and potential gains Pg .

Available things are: Xe(state error) , Pg(potential gains) .

Details of the meaning:

Resulting things are: G(guidance kinematics) .

D.3.4 Quaternion error

Sentences to use:

Determine quaternion error Qe between quaternion Qnow and quaternion Qdes .

Available things are: Qnow(quaternion) , Qdes(quaternion) .

Details of the meaning:

Resulting things are: Qe(quaternion error) .

D.3.5 Determine state error

Sentences to use:

Determine state error Xe between state Xnow and desired state Xdes .

Available things are: Xnow(state) , Xdes(desired state) .

Details of the meaning:

Define quaternion Qnow as "Xnow(7:10)" . Define quaternion Qdes as "Xdes(7:10)" . Determine quaternion error Qe between quaternion Qnow and quaternion Qdes . Execute "Xe=Xnow-Xdes;Xe(7:10)=Qe;" .

Resulting things are: Xe(state error) .

D.3.6 Saturate surface

Sentences to use:

Form modified joint sliding surface S2 by saturatating joint sliding surface S according to surface limit Sl .

Available things are: S(joint sliding surface) , Sl(surface limit) .

Details of the meaning:

Resulting things are: S2(joint sliding surface) .

D.3.7 Saturate control output

Sentences to use:

Formulate output control torque Tout and control force Fout by saturating control torque T and control force F with actuator limits Ubounds .

Available things are: T(control torque) , F(control force) , Ubounds(actuator limits) .

Details of the meaning:

Resulting things are: Tout(control torque) , Fout(control force) .

D.3.8 Return ideal state

Sentences to use:

Obtain current state Xnow from sensors observing current state X .

Available things are: X(current state) .

Details of the meaning:
Execute ”Xnow=X” .

Resulting things are: Xnow(current state) .

D.3.9 Define agent memory

Sentences to use:
Retrieve global memory M .

Details of the meaning:
M . SurfaceWeights = [0.1;0.1] ;M . SurfaceLimit = [0.1] ;M .

Resulting things are: M(global memory) .

D.3.10 Retrieve from memory

Sentences to use:
Retrieve ’name’ memory item OUT from global memory M .

Available things are: U_(quote) , M(global memory) .

Details of the meaning:
Execute ”N=fieldnames(M);” . Execute code ”index=find(ismember(N,U_)==1);” . Execute ”OUT=M.(Nindex);” .

Resulting things are: OUT(memory item) .

D.3.11 Perform continuous time regulatory smc

Sentences to use:
Use ’CTSMC’ to obtain control torque Tout and control force Fout for spacecraft agent SPA1 and desired state Xdes .

Available things are: U_(quote) , Xdes(desired state) , SPA1(spacecraft agent) .

Details of the meaning:
Retrieve global memory M . If U_ is ’CTSMC’ , then do the following . Determine control torque Tout and control force Fout based upon desired state Xdes and global memory M . Finish conditional actions .

Resulting things are: Tout(control torque) , Fout(control force) .

D.4 Trivia

D.4.1 Truth value of strings equal

Sentences to use:

Quote Q is 'quote' .

Available things are: Q(quote) , U_(quote) .

Details of the meaning:

Execute code "B=strcmp(Q,U_);" .

Resulting things are: B(relation Boolean) .

D.4.2 The special data types of numerical arrays, cell arrays and text with constraints

'code' which is text .
'control type' which is text .
'equation' which is text .
'image' which is text .
'input' which is an array of numbers .
'matlab code' which is text .
'matrix' which is an array of numbers .
'memory item' which is an array of numbers .
'number' which is an array of numbers .
'omega' which is an array of numbers .
'physical object' which is text .
'physical quantity' which is an array of numbers .
'position' which is an array of numbers .
'quaternion' which is an array of numbers .
'quaternion error' which is an array of numbers .
'quote' which is text .
'scalar' which is an array of numbers .
'sign threshold' which is an array of numbers .
'size' which is an array of numbers .
'spacecraft agent' which is text .
'spacecraft agent mass' which is an array of numbers .
'surface limit' which is an array of numbers .
'time period' which is an array of numbers .
'unit matrix' which is an array of numbers .
'vector' which is an array of numbers .
'velocity' which is an array of numbers .

D.4.3 Primitives: M-functions of hardware handling, signal processing and modelling

D.4.4 Ontology source text used

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ID:  AGENT ONTOLOGY FOR SPACECRAFT AGENT(S)
% Author : N K Lincoln
%
% School of Engineering Sciences
% University of Southampton (UK)
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%general defintions
>time period: physical quantity
>unit matrix : matrix
>matrix: double
>vector : double
>scalar : double
>physical object : char
>physical quantity: double
>size:double
>input: double
>cell array: vector

%The spacecraft agent itself:
>spacecraft agent : physical object
@agent name: char
@coordinate frames : set of char
@total mass : double
@inertial matrix: matrix
@output limitations: vector
@knowledge: cell array

>spacecraft agent mass: double
@dimension : {'kg', 'ton'}

>inertia matrix: matrix
@dimension : {'kgm^{-2}'}

>actuator limits: vector
```

```
@upper bounds: double
@lower bounds: double

%with knowledge
>global memory: cell array
>>memory item: matrix

%What it can do:
>spacecraft movements
@time axis : double
@rotations : set of quaternions
@translations : set of vectors

>spacecraft forces
@time axis:double
@forces: set of vectors

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Attributes of a spacecraft agent:

>desired state
@dimensions: set of char
@values : vector
@reference frame : char
@time horizon: double

>desired attitude
@dimension : {'Euler angles', 'quaternion'}
@values : vector
@reference frame : char

>desired position
@dimension : {'km','m','mm'}
@values : vector
@reference frame : char

>attitude error
@dimension : {'Euler angles', 'quaternion'}
@values : vector
@reference frame : char
```

```
>position error
@dimension : {'km','m','mm'}
@values : vector
@reference frame : char

>state : vector
>>position: vector
>>velocity: vector
>>quaternion: vector
>>quaternion error: vector
>>omega: vector

>state derivative: vector
>state error: vector
>current state: vector

>translational velocity : vector
@dimension: {'km/s','m/s'}
>translational velocity error : vector
@dimension: {'km/s','m/s'}
>angular velocity : vector
@dimension: {'rad/s'}
>angular velocity error : vector
@dimension: {'rad/s'}

%guidance inputs
>guidance kinematics: vector
>guidance omega gradient: vector
>guidance direction: vector
>guidance reference : vector

%control specific items
>surface weights : vector
>surface limit: double
>joint sliding surface: vector
>smoothed sign function: vector
>potential gains: vector

>control type: char
```

```
@CTSMC:char
@DTSMC: char
@regualtory:char
@model predictive: char


%spacecraft outputs
>dynamical force : matrix
@dimension: {'N','mN'}
>sign threshold : scalar
>control torque : vector
@dimension: {'Nm','mNm'}
>control force : vector
@dimension: {'Nm','mN'}


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```