Two-Stage Path Planning Approach for Designing
Multiple Spacecraft Reconfiguration Maneuvers
and Application to SPHERES onboard ISS

by

Georges Salim Aoudé

Bachelor of Engineering
McGill University, 2005

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2007

© Georges Salim Aoudé, MMVII. All rights reserved.

The author hereby grants to MIT permission to reproduce and
distribute publicly paper and electronic copies of this thesis document
in whole or in part.

Author

Department of Aeronautics and Astronautics

July 31, 2007

Certified by

Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by

Associate Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students
Two-Stage Path Planning Approach for Designing Multiple Spacecraft Reconfiguration Maneuvers and Application to SPHERES onboard ISS

by

Georges Salim Aoudé

Submitted to the Department of Aeronautics and Astronautics on July 31, 2007, in partial fulfillment of the requirements for the degree of Master of Science in Aeronautics and Astronautics

Abstract

The thesis presents a two-stage approach for designing optimal reconfiguration maneuvers for multiple spacecraft. These maneuvers involve well-coordinated and highly-coupled motions of the entire fleet of spacecraft while satisfying an arbitrary number of constraints. This problem is particularly difficult because of the nonlinearity of the attitude dynamics, the non-convexity of some of the constraints, and the coupling between the positions and attitudes of all spacecraft. As a result, the trajectory design must be solved as a single 6N DOF problem instead of N separate 6 DOF problems. The first stage of the solution approach quickly provides a feasible initial solution by solving a simplified version without differential constraints using a bi-directional Rapidly-exploring Random Tree (RRT) planner. A transition algorithm then augments this guess with feasible dynamics that are propagated from the beginning to the end of the trajectory. The resulting output is a feasible initial guess to the complete optimal control problem that is discretized in the second stage using a Gauss pseudospectral method (GPM) and solved using an off-the-shelf nonlinear solver. This thesis also places emphasis on the importance of the initialization step in pseudospectral methods in order to decrease their computation times. It demonstrates the improvement that an initial guess based on an RRT planner brings to an optimal control problem solved using pseudospectral methods. Finally, this thesis presents the successful results of several reconfiguration maneuver experiments performed using the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) hardware testbed onboard the International Space Station (ISS). The maneuvers were designed using two different two-stage algorithms presented in this work. It also discusses the lessons learnt from these tests, and the recommendations to improve future ISS reconfiguration experiments.

Thesis Supervisor: Jonathan P. How
Title: Professor of Aeronautics and Astronautics
Acknowledgments

There are several people that greatly contributed in this work. So I will try to thank them all here.

First of all, I would like to thank my advisor Prof. Jonathan How. His guidance and support were very essential to accomplish this work. His motivation to always push the limits in research was inspiring. I learnt a lot from Prof. How. I honestly owe him tremendously. I would also like to really thank Prof. David Miller for supporting me and giving me the opportunity to be part of the SPHERES team.

Thanks also to all my former and current labmates in the Aerospace Controls Laboratory who offered a lot of help during my masters thesis. In particular, I would like to thank Dr. Louis Breger, Dr. Yoshi Kuwata, Han-Lim Choi and Chun Sang Teo. Special thanks to Dr. Geoffrey Huntington for sharing his expertise with me when I needed it the most. Thanks also to my officemates for making the long hours spent in the office more enjoyable: Milan Mandic, Brandon Luders, Sungho Lee and Matt Jeffrey. I also would like to thank Ian Garcia who helped me get started on this work and gave me many valuable ideas afterwards. Another group of people that deserves special credit is the SPHERES team. I would especially like to thank Dr. Simon Nolet, Dr. Alvar Saenz-Otero, Dr. Hiraku Sakamot, Swati Mohan, Mark Hilstad, Nicholas Hoff, Chris Mandy and Amer Fejzic. I would also like to thank the following people from whom I learnt greatly by interacting with: Prof. Emilio Frazzoli, Dr. Umair Ahsun, Carl Nehme, and Prof. Anil Rao who was also very kind to let me use his latest sofware before its official release. Many thanks also to Kathryn Fisher and Peggy Bryan for their administrative help.

Finally, I would like to thank my dearest family: Mom, Dad, and my two sisters, Zeina and Georgette. I would not have done it without all your love and support. I am also very grateful to my uncle, Metropolitan Elias, for all his encouragement and prayers. Lastly, and most importantly, I would like to thank God, for anything good comes from Him.
I dictate this thesis to the memory of my grandfathers Georges and Aziz.

This research was funded under Payload Systems Inc. SPHERES Autonomy and Identification Testbed Grant #012650-001, NASA Space Communications Project Grant #NAG3-2839, and Le Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT) Graduate Award.
## Contents

1 Introduction .................................................. 19
   1.1 Problem Definition ........................................ 19
   1.2 Survey of Previous Work ................................... 22
   1.3 Solution Concepts ........................................ 24
      1.3.1 Two-Stage Path Planning ................................. 24
      1.3.2 Rapidly Exploring Random Trees ...................... 25
      1.3.3 Pseudospectral Methods ................................. 25
   1.4 ISS Experiments using SPHERES Testbed .................. 27
   1.5 Layout of Thesis .......................................... 27

2 Path Planning for Multiple Spacecraft Reconfiguration Maneuvers 31
   2.1 Nomenclature ............................................. 32
   2.2 Spacecraft Reconfiguration Problem Formulation .......... 33
   2.3 Solution Approach ........................................ 37
      2.3.1 The RRT First Stage ................................... 37
      2.3.2 The Augmentation with Feasible Dynamics .............. 39
      2.3.3 The GPM Second Stage .................................. 42
   2.4 Examples ................................................ 46
      2.4.1 Implementation Details ................................ 46
      2.4.2 Example: Two-Spacecraft Maneuver with Obstacle and Sun Avoidance ........................ 47
      2.4.3 Example: Coupled Two-Spacecraft Maneuver ............ 49
      2.4.4 Example: Four-Spacecraft Maneuver .................... 55
5 Experiments on SPHERES onboard ISS

5.1 SPHERES Background .................................................. 105
5.2 Reconfiguration Maneuvers on the SPHERES Testbed ................. 107
5.3 Flight Experiment Results .................................................. 108
  5.3.1 RRT-LS Experiment Results ........................................ 109
  5.3.2 RRT-LPM Experiment Results ...................................... 125
  5.3.3 Discussion ................................................................. 134
5.4 Summary ................................................................. 137

6 Conclusions and Future Work ............................................. 139

6.1 Conclusions ................................................................. 139
6.2 Recommendations for Future Work ...................................... 141
  6.2.1 Future Work for the RRT-GPM Approach .......................... 141
  6.2.2 Future Work for Real-Time Path Planning of Reconfiguration
      Maneuvers ................................................................. 141
List of Figures

1-1 Artist’s Impression of the TPF Mission .............................................. 20
1-2 The formation reconfiguration problem .............................................. 21
1-3 Example of an RRT expansion starting from the center of a square .... 26
1-4 A SPHERES micro-satellite onboard the ISS (Courtesy of NASA) .... 28

2-1 Example: Two-Spacecraft Maneuver with Obstacle and Sun Avoid-
ance. RRT Output. ................................................................. 48
2-2 Final trajectory of the two-spacecraft maneuver with obstacle and sun
avoidance example. ................................................................. 48
2-3 Distances between the center of the spacecraft and the center of the
fixed obstacle. ................................................................. 50
2-4 Cosine of the angles between the sensitive instrument of each spacecraft
and the sun vector. ................................................................. 50
2-5 Example: Two-Spacecraft Maneuver with Inter-Spacecraft Pointing
and Obstacle Avoidance. RRT Output. ......................................... 52
2-6 Final trajectory of the two-spacecraft maneuver with obstacle and
inter-spacecraft pointing. ................................................................. 52
2-7 Distances between centers of the spacecraft and center of obstacle lo-
caled at $[0.5, 0.7, 0.5]^T$ for the coupled two-spacecraft maneuver. .... 53
2-8 Distance between the centers of the spacecraft for the coupled two-
spacecraft maneuver. ................................................................. 53
2-9 Distances between centers of the spacecraft and center of obstacle lo-
cated at $[0.5, 0.1, 0.5]^T$ for the coupled two-spacecraft maneuver. .... 54
2-10 Cosines of angles between ranging device vector and relative position of both spacecraft for the coupled two-spacecraft maneuver.

2-11 Example: Four-Spacecraft Maneuver with Inter-Spacecraft Pointing and Absolute Pointing. RRT Output.

2-12 Final trajectory of the four-spacecraft maneuver with inter-spacecraft and absolute pointing.

2-13 Variation of Cost with Feasibility and Optimality Tolerances for a problem with RRT initial guess.

2-14 Variation of Computation Time with Feasibility and Optimality Tolerances for a problem with RRT initial guess.

2-15 Variation of Cost with Feasibility and Optimality Tolerances for a problem without RRT initial guess.

2-16 Variation of Computation Time with Feasibility and Optimality Tolerances for a problem without RRT initial guess.

2-17 Error in the final control of an optimal control problem with path constraints as a function of the number of nodes using three different interpolation techniques.

3-1 Steps Involved in Generating a Feasible Initial Guess to the Optimal Control Problem.

3-2 RRT Path and Final Trajectories for the Simple Single Spacecraft Example.

3-3 RRT Path and Final Trajectories for the Three-Spacecraft Example.

3-4 RRT Path and Final Trajectories for the Five-Spacecraft Example.

3-5 Comparison of computation times of a series of reconfiguration maneuvers of increasing size solved using a Gauss pseudospectral method.

3-6 Time Ratios of w/o RRT over with RRT Computation Times.

4-1 Final Trajectories for the single spacecraft maneuver.

4-2 States of the spacecraft over time in the solution of the single spacecraft problem using the RRT-GPM approach.
4-3 Controls of the spacecraft over time in the solution of the single spacecraft problem using the RRT-GPM approach. ....................... 90
4-4 States of the spacecraft over time in the solution of the single spacecraft using the RRT-LS approach. ............................... 91
4-5 Controls of the spacecraft over time in the solution of the single spacecraft using the RRT-LS approach. ............................... 91
4-6 Final Trajectories for the two-spacecraft maneuver. ............... 93
4-7 Controls of the spacecraft over time in the solution of the two-spacecraft using the RRT-GPM approach. ............................... 94
4-8 Controls of the spacecraft over time in the solution of the two-spacecraft using the RRT-LS approach. ............................... 95
4-9 Final Trajectories for the single spacecraft maneuver using the RRT-LPM approach. .................................................. 99
4-10 States of the spacecraft over time in the solution of the single spacecraft using the RRT-LPM approach. ............................... 100
4-11 Controls of the spacecraft over time in the solution of the single spacecraft using the RRT-LPM approach. ............................... 100
4-12 Final Trajectory for the two-spacecraft maneuver using the RRT-LPM approach. .................................................. 101
4-13 Controls of the spacecraft over time in the solution of the two-spacecraft using the RRT-LPM approach. ............................... 102

5-1 A SPHERES Microsatellite .................................................. 106
5-2 Two SPHERES performing a reconfiguration maneuver onboard the ISS using the RRT-LS approach (Courtesy of NASA). .......... 109
5-3 Snapshots from the video of a SPHERES performing a reconfiguration maneuver onboard the ISS using the RRT-LPM approach (Courtesy of NASA). .................................................. 110
5-4 Snapshots from the video of two SPHERES performing a reconfiguration maneuver onboard the ISS using the RRT-LPM approach (Courtesy of NASA). .................................................. 110
5-5 Theoretical final trajectory of the single spacecraft maneuver designed using the RRT-LS approach. ........................................... 112
5-6 Theoretical states of the spacecraft over time for the single spacecraft maneuver designed using the RRT-LS approach. ......................... 112
5-7 Theoretical input controls over time for the single spacecraft simulated maneuver designed using the RRT-LS approach. ......................... 113
5-8 Trajectory of the spacecraft during the ISS single spacecraft maneuver designed using the RRT-LS approach. ................................. 114
5-9 States of the spacecraft over time for the ISS single spacecraft maneuver designed using the RRT-LS approach. ................................. 114
5-10 State errors of the spacecraft over time for the ISS single spacecraft maneuver designed using the RRT-LS approach. ......................... 115
5-11 Input controls over time for ISS single spacecraft maneuver designed using the RRT-LS approach. .......................................... 115
5-12 Cosines of the angles between the ranging device and the restriction pointing vectors during the ISS single spacecraft maneuver designed using the RRT-LS approach. ........................................ 116
5-13 Distance between the center of the spacecraft and the center of the obstacle during the ISS single spacecraft maneuver designed using the RRT-LS approach. ........................................ 116
5-14 Theoretical trajectories of the two-spacecraft maneuver designed using the RRT-LS approach. ................................................. 117
5-15 Theoretical states of the spacecraft over time for the two-spacecraft maneuver designed using the RRT-LS approach. ......................... 118
5-16 Theoretical input controls for the spacecraft over time during the two-spacecraft maneuver designed using the RRT-LS approach. ........... 119
5-17 Trajectories of the spacecraft during the ISS two-spacecraft maneuver
designed using the RRT-LS approach ........................................ 120
5-18 States of the spacecraft over time for the two-spacecraft ISS maneuver
designed using the RRT-LS approach ......................................... 121
5-19 Input controls for the spacecraft over time during the two-spacecraft
ISS maneuver designed using the RRT-LS approach ......................... 122
5-20 State errors over time for the two-spacecraft spacecraft ISS maneuver
designed using the RRT-LS approach ........................................ 123
5-21 Cosines of angles between ranging device vector and relative position
of both spacecraft during the two-spacecraft ISS maneuver .............. 124
5-22 Distance between the centers of the spacecraft and the center of the
obstacle during the two-spacecraft ISS maneuver designed using the
RRT-LS approach ........................................................................... 124
5-23 Trajectory of the spacecraft during the ISS single spacecraft maneuver
designed using the RRT-LPM approach ....................................... 126
5-24 States of the spacecraft over time for the ISS single spacecraft maneuver
designed using the RRT-LPM approach ....................................... 126
5-25 State errors of the spacecraft over time for the ISS single spacecraft
maneuver designed using the RRT-LPM approach ......................... 127
5-26 Input controls over time for ISS single spacecraft maneuver designed
using the RRT-LPM approach ................................................ 127
5-27 Cosines of the angles between the ranging device and the restriction
pointing vectors during the ISS single spacecraft maneuver designed
using the RRT-LPM approach .................................................. 128
5-28 Distance between the center of the spacecraft and the center of the
obstacle during the ISS single spacecraft maneuver designed using the
RRT-LPM approach ........................................................................ 128
5-29 Trajectories of the spacecraft during the ISS two-spacecraft maneuver
designed using the RRT-LPM approach ..................................... 129
5-30 States of the spacecraft over time for the two-spacecraft ISS maneuver designed using the RRT-LPM approach.  
5-31 Input controls for the spacecraft over time during the two-spacecraft ISS maneuver designed using the RRT-LPM approach.  
5-32 State errors over time for the two-spacecraft spacecraft ISS maneuver designed using the RRT-LPM approach.  
5-33 Cosines of angles between ranging device vector and relative position of both spacecraft during the two-spacecraft ISS maneuver.  
5-34 Distance between the centers of the spacecraft and the center of the obstacle during the two-spacecraft ISS maneuver designed using the RRT-LPM approach.  
5-35 Conversion of the thrust commanded to thruster ON/OFF time with the use of the pulse-width modulator [1].
List of Tables

2.1 Comparison of computation times for problems with MRP versus same problems with quaternions ........................................... 60
3.1 Comparison of computation times of reconfiguration maneuvers for formation of increasing size solved using a Gauss pseudospectral method .................................................. 79
3.2 Comparison of final costs of reconfiguration maneuvers for formation of increasing size solved using a Gauss pseudospectral method .................................................. 82
5.1 Comparison of total costs of the ISS reconfiguration maneuvers against theoretical results. ........................................... 135
Chapter 1

Introduction

1.1 Problem Definition

The Terrestrial Planet finder (TPF) [2], the Laser Interferometer Space Antenna Project (LISA) [3], the Micro-Arcsecond X-ray Imaging Mission (MAXIM) [4], the System F6 Program to demonstrate a fractionated spacecraft approach [5], as well as many other future space missions and programs will be enabled by a formation flying technology for multiple spacecraft. Formation flying of spacecraft consists of more than one spacecraft whose dynamical states are coupled through a common control law [6]. For example, Figure 1-1 shows an artist’s impression of the proposed TPF observatory that consists of multiple spacecraft carrying infrared telescopes [2, 7]. The vehicles are independent, but they are coupled through the control objective of achieving a precise telescope.

Formation flying has been extensively investigated as a means to expand the capabilities of space missions focused on obtaining magnetosphere and radiation measurements, gravity field measurements, and 3-D mapping for planetary explorers (to name a few). The use of fleets of small satellites, instead of a single monolithic satellite, enables higher resolution imagery and interferometry, robust and redundant fault-tolerant spacecraft system architectures, and more complex networks of satellites, thereby improving science return. To achieve these benefits, tighter requirements will be imposed on the communication and coordination between spacecraft, path
Figure 1-1: Artist’s Impression of the TPF Mission [2, 7]

planning algorithms, autonomous fault detection and recovery, and on the high level mission management [8].

There are two key types of trajectory design problems for formation flying spacecraft: 1) reconfiguration, which consists of maneuvering a fleet of spacecraft from one formation to another, and 2) station-keeping, which consists of keeping a cluster of fleet of spacecraft in a specific formation for a determined part of the trajectory. Both types of formation flying maneuvers must be addressed for deep-space missions where the relative spacecraft dynamics usually reduces to double integrators, or planetary orbital environment flying missions where spacecraft are subjected to significant orbital dynamics and environmental disturbances [9].

This thesis focuses on the trajectory design of reconfiguration maneuvers of multiple spacecraft in deep space environment. They consist of moving and rotating a group of \( N \) spacecraft from an initial configuration to a desired final configuration, while satisfying different types of constraints (see Figure 1-2). These constraints may consist of collision avoidance, restrictions on the region of the sky where certain spacecraft instruments can point (e.g., a sensitive instrument that cannot point at the Sun), or restrictions on pointing towards other spacecraft (e.g., requirements on maintaining inter-spacecraft communication links and having cold science instruments avoid
high temperature components on other vehicles).

It is also desirable to optimize some performance index (fuel, energy, maneuver time, etc.) [9]. This problem is particularly difficult because of the nonlinearity of the attitude dynamics, the non-convexity of some of the constraints, and the coupling between the positions and attitudes of all spacecraft. Even though several solutions exist for the attitude control problem alone, its intrinsic complexity, arising from its nonlinearity, makes the general spacecraft reconfiguration problem harder. The non-convex constraints place this problem in a general class of path planning problems with a computational complexity that is exponential in the number of degrees of freedom of the problem. In addition, some types of pointing constraints forces coupling between the position and attitude the spacecraft, making it impossible to separate the translation control problem from the attitude control problem. As a result, the trajectory design must be solved as a single 6N DOF problem instead of N separate 6 DOF problems. Since the size of future formation flight missions will continue to increase [11], new path planning techniques should be able to handle large scale formations. Another requirement is that these planning algorithms should have fast
computation times suitable for an eventual online implementation.

1.2 Survey of Previous Work

The spacecraft trajectory design of the constrained and unconstrained translation and attitude maneuvers has been the subject of extensive research in formation flying spacecraft. Most of the solutions for this problem consider either the translation or the attitude trajectories.

Refs. [12, 13] proposed the use of Mixed Integer Linear Programming (MILP) or Mixed Integer Linear Matrix Inequalities (MI/LMI) techniques to solve the problem. These methods require several simplifications in formulating the problem. MILP deals with linear problems, therefore the systems dynamics as well as the constraints should be represented in linear form. A major drawback of the MI/LMI solution technique is that the size of the problem increases dramatically with the number of spacecraft, whereas solving a MILP problem usually requires branch and bound techniques.

Some authors considered the use of potential functions in the solution of path planning problems [14, 15]. The major drawback of this type of methods is that the trajectory it generates might get trapped in local minima. Moreover, computing a potential function that is free of local minima is computationally very hard for any non-trivial set of constraints [16]. This approach cannot either guarantee that the resulting trajectories are collision free, which is critical in spacecraft formation flying missions.

Another popular approach that has been investigated recently with great success in motion planning research is the use of randomized motion planning algorithms such as the probabilistic roadmap (PRM) planners [17, 18], and their incremental counterparts the Rapidly-exploring Random Trees (RRT) algorithms (see Section 1.2.2). But the application of RRTs on spacecraft reconfiguration problems was limited to 1) a problem involving a single spacecraft with no pointing constraints [19], 2) a problem considering the attitude maneuver of one spacecraft [16], and 3) a multi-spacecraft reconfiguration problem that solves for the translation trajectory only [20]. It’s only
very recently that the more general case of combined translation and attitude reconfiguration of multiple spacecraft problems has been addressed using RRTs \cite{21, 22}. This approach consists of a two-stage planning algorithm, similar to the one developed in this thesis. However, its second stage, also called the “smoothing” step, is based on linearizing a nonlinear optimization problem around the feasible solution generated by the first stage. This induces linearization errors in the solution of the problem, which can make it infeasible. Additional work is needed to restore feasibility, and it is problem dependent.

Numerous researchers have recently explored using pseudospectral methods for nonlinear trajectory optimization problems related to aerospace applications \cite{23–25}. One major negative aspect of pseudospectral methods in general is that the computation time increases dramatically with the complexity of the problem. There are many possible reasons for this increase, but one noticeable issue is that simply finding a feasible solution to a problem as complex as the multi-spacecraft reconfiguration problem set of solutions. Providing a feasible initial guess to the solver should help decrease this computation time, but this is complex since the path planning problem with general constraints is NP-hard \cite{26}. Ref. \cite{24} suggests using a “warm start” to improve the computation times of the problem. A warm start considers the solution of previous optimizations as an initial guess to the current problem. This idea is similar to the mesh refinement technique introduced in Ref. \cite{27}, which starts with a coarse grid (i.e. low number of discretization nodes), and if necessary, refines the discretization, and then repeats the optimization steps. But these approaches can be very time consuming, and therefore not feasible for online planning of reconfiguration maneuvers. If a warm start is to be efficient, the algorithm must be chosen with care. This thesis presents a technique for improving the performance of the pseudospectral based solution by providing a feasible initial guess. This guess is the solution of a simplified version of the path planning problem without differential constraints. This problem is quickly solved using an improved version of bi-directional RRTs \cite{22}. 

23
1.3 Solution Concepts

This thesis presents a two-stage path planning algorithm to solve the problem discussed in Section 1.1. This technique is briefly introduced in Section 1.3.1, and explained further in Chapter 2. The first stage, discussed in Section 1.3.2, is based on Rapidly-exploring Random Trees (RRTs), a randomized planning technique that has been very popular recently. Then, Section 1.3.3 discusses the use of pseudospectral methods as a solution technique for the optimal control problem formed in the second stage of the path planning algorithm. This two-stage technique extends the original ideas in Ref. [10] to improve the second step by using a specific pseudospectral method, called the Gauss pseudospectral method (GPM).

1.3.1 Two-Stage Path Planning

The constraints encountered in spacecraft reconfiguration maneuver problems fall into two main categories, (a) kinematic and (b) dynamic. Kinematic constraints address the motion of the spacecraft under consideration, but ignore the forces behind the motion, which are captured in the dynamic constraints. Path planning for reconfiguration maneuvers is a challenging task even when considering each set of constraints individually. When addressing these two types of constraints simultaneously, the problem is known as kinodynamic motion planning [28], which has been traditionally implemented using two common approaches: “two-stage” planning and “state-space” formulation [28, 29]. Unlike the “state-space” approach, where the dynamic constraints are taken into account from the start of the algorithm [30, 31], the “two-stage” formulation consists of first finding a feasible path that satisfies the kinematic constraints, and then optimizing this path to include the dynamic constraints [21, 32]. This thesis develops a two-stage approach for solving reconfiguration maneuvers of multiple spacecraft. Examples of complex maneuvers including up to five spacecraft illustrate this approach.
1.3.2 Rapidly Exploring Random Trees

The first stage of the two-stage algorithm developed in this thesis concentrates on finding any feasible trajectory for the problem, postponing the "smoothing" or cost improvement to the second stage. However, finding a feasible path with guarantees is by itself very difficult because the path planning problem becomes intractable for high dimensional problems like the multiple spacecraft reconfiguration maneuver problem. But it has been shown that if the guaranteed completion is relaxed, larger problems can be solved using randomized path planning algorithms, such as the Probabilistic Roadmaps (PRMs) [17]. Rapidly exploring Random Trees (RRTs), a recent variant of PRMs introduced in Refs. [33, 34], was developed for planning under differential constraints, but it has been applied mostly in ordinary motion planning. The RRT structure and algorithm are designed to efficiently explore high-dimensional spaces, therefore quickly finding a feasible solution even in highly constrained environments.

RRTs have several nice properties [33]. We emphasize two of them: 1) their expansion is heavily biased towards unexplored areas of the configuration space (e.g., see Figure 1-3) and 2) the RRT algorithm is probabilistically complete i.e., the probability of finding a feasible path approaches one as the number of iterations increases. RRTs and their variants have been applied successfully in several applications in different areas of research including robotics and graphics [35]. This thesis uses an improved version of the well known bidirectional RRTs, a technique that has been introduced and shown to be a very fast planner for trajectory optimization problems when differential constraints are ignored [22]. Section 2.3.1 describes this method in more detail.

1.3.3 Pseudospectral Methods

The second stage of the planning algorithm developed in this thesis is formulated as an optimal control problem with path constraints. Numerical methods for solving this type of problems fall into two general categories: direct methods and indirect methods [36].
In an indirect method, the optimal solution is found by solving a Hamiltonian boundary-value problem derived from the first-order necessary conditions for optimality. The primary advantages of indirect methods are their high accuracy in the solution and the assurance that the solution satisfies the first-order optimality conditions. However, indirect methods have several disadvantages including possible difficulties in deriving the Hamiltonian boundary-value problem, small radii of convergence, and the requisite of a good initial guess for both the state and costate.

In a direct method, the continuous-time optimal control problem is transcribed to a nonlinear programming problem (NLP). The resulting NLP can be solved by well developed algorithms and software. Direct methods have the advantage that the optimality conditions do not need to be derived. They do suffer however, depending on the type of direct method, in that the solution may not contain any costate information, or may result in an inaccurate costate.

As the number of spacecraft in the reconfiguration problem increases, solving the Hamiltonian boundary value problem becomes increasingly difficult, if not impossible. Moreover, advances in direct methods, such as the pseudospectral methods [37, 38], have improved the accuracy of the costate information compared to earlier direct methods.

The states and controls in pseudospectral methods are parameterized using a basis of global polynomials which are derived from an appropriate set of discretization points [39]. The use of global orthogonality makes it simple to transform the original
problem into a set of algebraic equations. The discretized optimal control problem is then transcribed to a nonlinear program which can then be solved using an off-the-shelf nonlinear solver. This thesis uses the Gauss pseudospectral method (GPM), one of the newest numerical approaches in the literature today, that has shown promise both in the solution and in the post-analysis optimality \cite{23, 40}. Section 2.3.3 describes the Gauss pseudospectral method in its most current form.

1.4 ISS Experiments using SPHERES Testbed

This thesis also presents recent results of reconfiguration maneuvers experiments performed onboard the International Space Station (ISS) in March and April (2007) using the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) hardware testbed \cite{41, 42}. SPHERES is part of the Space Systems Laboratory (SSL) at MIT. Figure 1-4 shows a SPHERES micro-satellite during one of the tests performed onboard ISS. The reconfiguration maneuvers were designed using two different algorithms: 1) a two-stage algorithm developed in this thesis, and 2) another path planning algorithm that can be found in Ref. \cite{10}. Both algorithms were developed in the Aerospace Control Laboratory (ACL) at MIT. The experiments allowed to test the reconfiguration algorithms in real microgravity environment, thus validating the theoretical results. Chapter 5 describes the different experiments performed, shows graphs and images, analyzes the results of the tests, and gives the lessons learnt from these experiments.

1.5 Layout of Thesis

Chapter 2 describes the reconfiguration maneuver problem and the steps of the algorithm developed to solve it. It illustrates this new algorithm with several examples of increasing complexity, and discusses the implementation issues in solving these problems. This chapter contains a major contribution of this research: the extension of the two-stage approach developed in Ref. \cite{10} to improve the second stage by for-
mulating it as a problem based on a pseudospectral method, and solving it using a state-of-the-art pseudospectral method solver. This chapter also describes a transition algorithm that ensures that feasibility is maintained in the transition between the first and second stage.

Chapter 3 investigates the importance of the first step based on RRT on improving the performance of the second step based on Gauss pseudospectral methods. Several examples of reconfiguration maneuvers of increasing complexity illustrate this improvement. This chapter includes another main contribution of this work: the extension of the two stage solution approach developed in Ref. [10] to consider a tight integration with a pseudospectral solver. This integration is shown to considerably decrease the computation times of the solution process of problems based on pseudospectral methods.

Chapter 4 draws a comparison between the main two-stage algorithm developed in this thesis against two other algorithms: 1) another two-stage algorithm that this thesis develops, where the second stage is based on Legendre Pseudospectral methods, and 2) a different two-stage algorithm presented in [10], where the second stage is
formulated as a linear program.

Chapter 5 presents and analyzes the results of the implementation of the reconfiguration maneuver algorithms presented in Chapter 4 on the SPHERES testbed onboard the ISS. The value of this contribution lies in the implementation of these algorithms in a microgravity environment similar to that of real space missions, and the comparison of the flight data with the theoretical results enabling the validation of the designed algorithms.
Chapter 2

Path Planning for Multiple Spacecraft Reconfiguration Maneuvers

This chapter outlines the solution technique developed to solve the multiple spacecraft reconfiguration maneuver problem with combined translation and attitude. These problems are typically minimum fuel/energy, fixed-time maneuvers, and may include collision avoidance and pointing constraints. The solution technique is based on solving the problem in two stages: the first stage quickly provides an initial guess using the RRT planner and a transition algorithm that augments the guess with feasible dynamics, and the second stage incorporates this guess into the full optimal control problem that is discretized using the Gauss pseudospectral method (GPM) and solved using an off-the-shelf nonlinear solver [43]. The two-stage algorithm will be referred to as the RRT-GPM algorithm.

Section 2.2 introduces the problem formulation of spacecraft reconfiguration maneuvers.

In Section 2.3, the two-stage path planning solution approach will be introduced. There will be a review of the improved version of the bi-directional RRT algorithm [22] and the gauss pseudospectral method [39]. Furthermore, the transition algorithm will be introduced.
Section 2.4 includes several demonstrations of increasing difficulties of the two-stage algorithm, including uncoupled and coupled maneuvers.

Section 2.5 discusses implementation issues of the RRT-GPM algorithm and introduces methods to solve them.

## 2.1 Nomenclature

This chapter uses the following variable names:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r )</td>
<td>spacecraft position</td>
</tr>
<tr>
<td>( \dot{r} )</td>
<td>spacecraft velocity</td>
</tr>
<tr>
<td>( \sigma )</td>
<td>spacecraft attitude modified Rodrigues parameters</td>
</tr>
<tr>
<td>( w )</td>
<td>spacecraft angular velocity</td>
</tr>
<tr>
<td>( M )</td>
<td>mass of spacecraft</td>
</tr>
<tr>
<td>( J )</td>
<td>inertial matrix of spacecraft</td>
</tr>
<tr>
<td>( f )</td>
<td>input force</td>
</tr>
<tr>
<td>( \tau )</td>
<td>input torque</td>
</tr>
<tr>
<td>( x )</td>
<td>spacecraft state (contains ( r, \dot{r}, \sigma ) and ( w ))</td>
</tr>
<tr>
<td>( u )</td>
<td>control inputs (contains ( f ) and ( m ))</td>
</tr>
<tr>
<td>( l )</td>
<td>obstacle position</td>
</tr>
<tr>
<td>( p )</td>
<td>point of trajectory</td>
</tr>
<tr>
<td>( t )</td>
<td>time</td>
</tr>
<tr>
<td>( T )</td>
<td>total maneuver time</td>
</tr>
<tr>
<td>( N )</td>
<td>number of spacecraft</td>
</tr>
<tr>
<td>( N' )</td>
<td>number of discretization nodes</td>
</tr>
</tbody>
</table>

It also uses the following subscripts:

<table>
<thead>
<tr>
<th>Subscript</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i, j )</td>
<td>spacecraft</td>
</tr>
<tr>
<td>( k )</td>
<td>pointing constraints</td>
</tr>
<tr>
<td>( l )</td>
<td>discretization nodes</td>
</tr>
<tr>
<td>( o )</td>
<td>obstacles</td>
</tr>
</tbody>
</table>
2.2 Spacecraft Reconfiguration Problem Formulation

The general reconfiguration problem resides in finding a trajectory of $N$ spacecraft from time 0 to time $T$. Let $p_i(t)$ be a point of the trajectory of a single spacecraft at time $t$. This point consists of

$$p_i(t) = [x_i(t), u_i(t)], \quad (2.1)$$

where $x_i(t)$ and $u_i(t)$ represent the state and control inputs at each time $t$, respectively,

$$x_i(t) = [r_i(t), \dot{r}_i(t), w_i(t), \sigma_i(t)], \quad (2.2)$$

$$u_i(t) = [f_i(t), \tau_i(t)], \quad (2.3)$$

and where $i \in 1 \ldots N$ indicates the spacecraft. $r_i(t) \in \mathbb{R}^3$ is the position of its center, $\dot{r}_i(t) \in \mathbb{R}^3$ is its velocity, $w_i(t) \in \mathbb{R}^3$ its angular velocity, and $\sigma_i(t) \in \mathbb{R}^3$ is its attitude representation in modified Rodrigues parameters (MRP) [44]. All these variables are measured with respect to a local inertially fixed frame. The choice of MRP over other attitude representations is discussed in section 2.5.1. $f_i(t) \in \mathbb{R}^3$ represents the control input force, and $\tau_i(t) \in \mathbb{R}^3$ the control input torque. Therefore,

$$p(t) = [\ldots, p_i(t), \ldots], \quad (2.4)$$

represents a point in the composite trajectories of all the spacecraft at time $t$. Since the interest of this research is in deep space missions, the translation dynamics are approximated with a simple double integrator

$$\begin{bmatrix} \dot{r}_i(t) \\ \ddot{r}_i(t) \end{bmatrix} = \begin{bmatrix} 0_{3 \times 3} & I_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} \end{bmatrix} \begin{bmatrix} r_i(t) \\ \dot{r}_i(t) \end{bmatrix} + \begin{bmatrix} 0_{3 \times 3} \\ \frac{I_{3 \times 3}}{M} \end{bmatrix} f_i(t), \quad (2.5)$$
where $M \in \mathbb{R}$ is the mass, assumed to be the same for all spacecraft for simplicity. $I_{3 \times 3}$ is the $3 \times 3$ identity matrix, and $0_{3 \times 3}$ is the $3 \times 3$ zero matrix.

The attitude dynamics in MRP notation of the spacecraft considered as a rigid body are

$$
\dot{\sigma}_i(t) = R(\sigma_i(t))w_i(t) \quad (2.6)
$$

$$
J \dot{w}_i(t) = -w_i(t) \times Jw_i(t) + \tau_i(t) = -S(w_i(t))Jw_i(t) + \tau_i(t) \quad (2.7)
$$

where $J \in \mathbb{R}^{3 \times 3}$ is the spacecraft constant inertia matrix, considered to be the same for all spacecraft for simplicity. $S \in \mathbb{R}^{3 \times 3}$ is the skew-symmetric matrix representing the cross product operation

$$
S(a) \triangleq [a \times] = \begin{bmatrix}
0 & -a_3 & a_2 \\
a_3 & 0 & -a_1 \\
-a_2 & a_1 & 0
\end{bmatrix}, \quad \forall a \in \mathbb{R}^3 \quad (2.8)
$$

The Jacobian matrix $R \in \mathbb{R}^{3 \times 3}$ for MRP attitude representation is given by [44]

$$
R(\sigma_i(t)) = \frac{1}{4} \left[ (1 - \sigma_i^T(t)\sigma_i(t))I_{3 \times 3} + 2S(\sigma_i(t)) + 2\sigma_i(t)\sigma_i^T(t) \right] \quad (2.9)
$$

The path constraints can be divided into two categories: 1) collision avoidance constraints, and 2) pointing restriction constraints.

The collision avoidance category contains the inter-spacecraft collision avoidance constraints, which ensure safe separation between every pair of spacecraft, and are written as

$$
\|r_i(t) - r_j(t)\| \geq R_{ij} \quad (2.10)
$$

for $i, j \in 1 \ldots N, i \neq j$, and $R_{ij}$ is the minimum distance allowed between the centers of spacecraft $i$ and $j$. Collision avoidance also contains the obstacle avoidance constraints, which ensure safe maneuvering of every spacecraft among all obstacles,
and are written as

$$\| \mathbf{r}_i(t) - l_o(t) \| \geq R_{io}$$  \hfill (2.11)$$

for every obstacle \( o \), and for \( i \in 1 \ldots N \). \( l_o \) is the position of the center of obstacle \( o \), and \( R_{io} \) is the minimum distance allowed between the centers of spacecraft \( i \) and obstacle \( o \).

The pointing restriction category contains four types of constraints:

- Absolute stay outside constraints
- Absolute stay inside constraints
- Relative stay outside constraints
- Relative stay inside constraints

The absolute stay outside constraints can be written as

$$z_k^T y_k(t) \leq \cos \theta_k$$  \hfill (2.12)$$

for every stay outside pointing constraint \( k \). This constraint ensures that the spacecraft vector \( y_k \) remains at an angle greater than \( \theta_k \in [0, \pi] \) from the inertial vector \( z_k \). The vector \( y_k \) represents the body vector \( y_{kB} \) in the inertial coordinate frame.

The transformation of coordinates is given by

$$y_k(t) = \text{Rot}^{-1}(\sigma(t)) y_{kB}$$  \hfill (2.13)$$

where \( \text{Rot}(\sigma(t)) \) is the rotation matrix representation of the MRP attitude vector \( \sigma(t) \), which can be written as \[44\]

$$\text{Rot}(\sigma(t)) = I + \frac{4(1 - \sigma^T(t)\sigma(t))}{(1 + \sigma^T(t)\sigma(t))^2} S(\sigma(t)) + \frac{8}{(1 + \sigma^T(t)\sigma(t))^2} S(\sigma(t))^2$$  \hfill (2.14)$$

where \( S \) is the matrix defined in (2.8). It is assumed that \( y_{kB} \) and \( z_k \) are fixed vectors \( i.e. \), independent of time \( t \).
The absolute stay inside constraints only change the sign of the inequality of (2.12). They can be written as

\[ z_k^T y_k(t) \geq \cos \theta_k \]  

(2.15)

The inter-spacecraft relative stay outside constraints are given by

\[ \hat{r}_{ij}^T(t) y_k(t) \leq \cos \theta_k \]  

(2.16)

where \( y_k(t) \) and \( \theta_k \) are the same as defined above, and

\[ \hat{r}_{ij}(t) = \frac{r_j(t) - r_i(t)}{\| r_j(t) - r_i(t) \|} \]  

(2.17)

represents the unit vector pointing from spacecraft \( i \) to spacecraft \( j \). The inter-spacecraft relative stay inside can be similarly written as

\[ \hat{r}_{ij}^T(t) y_k(t) \geq \cos \theta_k \]  

(2.18)

The boundary conditions specify the initial and final configuration \( i.e., \) state of each spacecraft. They can be written as

\[ x_i(0) = x_{is} \]  

(2.19)

\[ x_i(T) = x_{if} \]  

(2.20)

where \( x_{is} \) represents the state corresponding to the specified starting condition, and \( x_{if} \) to the specified final condition \( \forall i \in 1 \ldots N \).

The state and control vectors are restricted to lie within specified bounds

\[ x_{\min} \leq x_i(t) \leq x_{\max} \]  

(2.21)

\[ u_{\min} \leq u_i(t) \leq u_{\max} \]  

(2.22)

where the inequality is understood to be component wise. The bounds on the input
control vectors are usually due to the limited thrust of each spacecraft. The bounds on the velocity vectors are usually characteristic of safety limits. Finally, the position bounds ensure that the problem space is bounded [45].

The objective is to minimize the total energy of the formation

\[ J = \sum_{i=1}^{N} \int_{0}^{T} \| f_i(t) \|^2 + \| \tau_i(t) \|^2 dt \]  \hspace{1cm} (2.23)

Minimizing the total energy consumption of a formation of spacecraft is an objective for many space missions [46, 47]. Furthermore, the energy is in general directly related to the fuel consumption. So minimizing energy leads to less fuel consumption.

2.3 Solution Approach

2.3.1 The RRT First Stage

The first stage of the RRT-GPM algorithm is based on the improved version of the bidirectional rapidly-exploring random trees (RRT) developed by the authors of Ref. [22]. It is reproduced here for clarity. The original RRT algorithm was developed by Lavalle [35].

In Algorithm 2.1, \( T_a \) and \( T_p \) represent trees having a composite trajectory point \( p \) at each node (2.4). \( T_a \) starts from the initial point and \( T_b \) starts from the final point of the goal trajectory. At each node, the points \( p \) are considered at rest, so the position and attitude are the only information of interest in this algorithm. At each iteration, \( \alpha(i) \) generates a random point, and then the point in the tree \( T_a \) with the minimum distance to the point \( \alpha(i) \) is found by calling Nearest(\( T_a, \alpha(i) \)). Distance in this context represents a weighted summation of rotation and translation. It can be written as

\[ d(p_1, p_2) = \sum_{i=1}^{N} \| r_{1,i} - r_{2,i} \| + K_a \angle(q_{1,i}, q_{2,i}) \]  \hspace{1cm} (2.24)

where \( \angle(q_{1,i}, q_{2,i}) \) represents the angle of an eigen-axis rotation between attitude \( q_{1,i} \).
Algorithm 2.1 RRT-BIDIRECTIONAL \((p_i, p_f)\)

1: \(T_a.\text{init}(p_i); T_b.\text{init}(p_f);\)
2: for \(j \leftarrow 1\) to \(K\) do
3: \(p_n \leftarrow \text{Nearest}(T_a, \alpha(j))\)
4: \(p_s \leftarrow \text{Potential-Connect}(p_n, \alpha(j))\)
5: if \(p_s \neq p_n\) then
6: \(T_a.\text{Add-Vertex}(p_s)\)
7: \(T_a.\text{Add-Edge}(p_n, p_s)\)
8: \(\hat{p}_n \leftarrow \text{Nearest}(T_b, p_s)\)
9: \(\hat{p}_s \leftarrow \text{Potential-Connect}(\hat{p}_n, p_s)\)
10: if \(\hat{p}_s \neq \hat{p}_n\) then
11: \(T_b.\text{Add-Vertex}(\hat{p}_s)\)
12: \(T_b.\text{Add-Edge}(\hat{p}_n, \hat{p}_s)\)
13: end if
14: if \(\hat{p}_s = p_s\) then
15: return Solution
16: end if
17: end if
18: \(\text{Swap}(T_a, T_b)\)
19: end for
20: return Failure

and \(q_{2,i}\) for spacecraft \(i\), and \(K_a\) is a weight factor that relates the translation distance and rotation angle.

Potential-Connect is an artificial potential function based on a distance metric \(d(p_1, p_2)\) where the obstacle avoidance, restricted pointing, and other constraints are represented by inequality and equality constraints \([22]\). So Potential-Connect is a search algorithm that tries to find a sequence of feasible points with a decreasing distance to the target point. This search can be formulated as a nonlinear optimization problem Algorithm 2.2. The solution to this problem can be found using a feasible sequential optimization method, and thus guarantees that the sequence of points represent a valid trajectory.

Potential-Connect tries to connect \(p\) to \(p_f\) by moving in small \(dp\) increments. These \(dp\) increments are restricted to be smaller in norm than \(\epsilon\) to guarantee feasibility between adjacent points of the trajectory. Note that the numerical experiments were done using a custom sequential linear solver that computes the solution of a sequence of linear programs with linearized constraints.
Algorithm 2.2 Potential-Connect($p$, $p_f$)

1: for $j \leftarrow 1$ to $K$ do
2:    Solve nonlinear program:
3:        \[
4:        \min_{d p} d(p + dp, p_f) \in \mathcal{P} \kappa_{min,i} \leq g_i(p + dp) \leq g_{max,i}, \forall i
5:        \|dp\| \leq \epsilon
6:    \]  
7:    End of nonlinear program
8:    $p \leftarrow p + dp$
9: end for
10: return $p$

So the solution of the first stage consists of a sequence of points from the initial point $p_i$ to the final point $p_f$. At each point, the spacecraft are assumed to be at rest, and there exists a direct motion to the next point that is guaranteed to satisfy all the constraints.

This improved bidirectional RRT planner has been demonstrated to be significantly faster than other similar spacecraft reconfiguration maneuver planners. For more details and illustrations of Algorithm 2.1 and Algorithm 2.2, the interested reader is encouraged to consult references [22] and [10].

2.3.2 The Augmentation with Feasible Dynamics

A major simplification in the first stage of the RRT-GPM algorithm is based on ignoring the differential constraints of the spacecraft reconfiguration problem. This simplification is essential in decreasing the computation time of the first stage. The RRT solution of the first stage is clearly suboptimal since the spacecraft are assumed to be at rest at each of the nodes, and no cost function is actually optimized. Therefore, a second stage is needed to improve the cost of the trajectory. A transition step that augments the RRT solution with feasible dynamics is thus required to allow using this solution as a feasible initial guess to the second stage. So a main requirement of this transition step is to ensure that feasibility is maintained between the first and second stage.

The idea of this transition step starts with adding an intermediate node $p^{inter}$
half way between every pair of nodes. First assume that the problem consists of only one spacecraft. To propagate the dynamics to \( p^{\text{inter}} \), the spacecraft is assumed to accelerate under a constant input force \( \hat{f} \) and a constant input torque \( \hat{\tau} \). \( \hat{f} \) and \( \hat{\tau} \) are chosen such that they satisfy the bounds on the forces and torques defined in (2.22). Once the spacecraft reaches \( p^{\text{current}} \), it decelerates under a constant force \(-\hat{f}\) and a constant torque \(-\hat{\tau}\) until it stops at the next node \( p^{\text{next}} \). This is a simple way to guarantee that the controls of the spacecraft are satisfied along each consecutive nodes. The smaller \( \hat{f} \) and \( \hat{\tau} \) are, the longer the total maneuver time is. Therefore, these values should be chosen to also satisfy the design specifications (e.g., total maneuver time) of the reconfiguration maneuver. Note that the restriction that the intermediate node lies exactly in between the original pair of nodes only exists in the initial guess. After the initial guess is given to the GPM stage, that restriction disappears, along with the assumption of fixed forces and torques.

To expand this idea to multiple spacecraft, it is necessary that all spacecraft reach the intermediate node \( p^{\text{inter}} \) at the same instant of time. This synchronization assumption is a simple way to ensure the feasibility of the algorithm when considering multiple spacecraft. But again, this assumption is relaxed after the guess is given to the second stage, i.e., the final solution of the RRT-GPM approach is not required to satisfy it. First of all, \( t_{\text{max}} \) is computed. \( t_{\text{max}} \) represents the maximum time needed by all spacecraft to reach \( p^{\text{inter}} \), if they all move under the same constant force \( \hat{f} \) and rotate under the same constant torque \( \hat{\tau} \). Then fixing the time to reach \( p^{\text{inter}} \) to be \( t_{\text{max}} \) for all spacecraft, the constant forces and torques responsible to move each spacecraft are recomputed. Therefore some spacecraft will be designed to move under forces smaller than \( \hat{f} \), and torques smaller than \( \hat{\tau} \).

Algorithm 2.3 contains the main steps of the transition algorithm. \( \text{traj} \) is the RRT output of the first stage, and \( p^k \) represents a point in the composite trajectories of all the spacecraft at node \( k \) (2.4). To make the algorithms simpler, it is assumed that the unknown values of the RRT output i.e., velocities, forces and torques, are all set to zero by default. An Update operation sets these variables to the correct values, and a Propagate operation creates new nodes using information of existing
Algorithm 2.3 Augment-Trajectory(traj)

1: augmented-traj ← ∅
2: for k ← 1 to Size(traj)−1 do
3:    augmented-traj ← Append(augmented-traj, Propagate-Dynamics(p^k, p^{k+1}))
4: end for
5: augmented-traj ← Append(augmented-traj, Update-Node(p^{end}))
6: return augmented-traj

updated nodes. Lines 2-4 in Algorithm 2.3 propagates the dynamics between each consecutive pairs of nodes of the trajectory. Line 5 ensures that the dynamics of the last node of the trajectory is also updated. Algorithm 2.4 describes how the dynamics are propagated between each pair of nodes. Line 1 calls Get-Max-Time to compute \( t_{max} \). Refer to Algorithm 2.5 for the steps involved in Get-Max-Time. Continuing with Algorithm 2.4, line 3 updates the forces and torques of the current node \( p^{current} \), which are ensured to be feasible by construction. The velocities at \( p^{current} \) remain zeros to satisfy the assumption of the first stage algorithm. Line 4 propagates the dynamics to the intermediate node \( p^{inter} \) by using the logic explained earlier in this section.

Note that the transition step has to also ensure that the velocities of the spacecraft always lie between the permissible bounds. This check can be incorporated easily in the Get-Max-Time function, but it is not usually a dominating factor. The reason is that, in general, the main limitation of the spacecraft is the maximum thrust it can exert, not the maximum velocity it can reach. The transition step runs in \( O(Nn) \), where \( N \) is the size of the formation, and \( n \) is the total number of nodes in the RRT output.
In summary, the transition step is a technique that augments the output of the RRT output of the first stage of the RRT-GPM algorithm with feasible dynamics. It consists of adding intermediate nodes with a complete specific set of feasible dynamics, and synchronizes all spacecraft to get to each node at the same time with feasible forces and torques. The whole process can be easily automated.

**Algorithm 2.5 Get-Max-Time(\(p^{\text{current}}, p^{\text{next}}\))**

1: \(t_{\text{max}} \leftarrow 0\)
2: for \(i \leftarrow 1\) to \(N\) do
3: \(t_{\text{max}} \leftarrow \text{Max}(t_{\text{max}}, \text{Get-Max-Rotation-Time}(p^{\text{current}}_i, p^{\text{next}}_i))\)
4: \(t_{\text{max}} \leftarrow \text{Max}(t_{\text{max}}, \text{Get-Max-Translation-Time}(p^{\text{current}}_i, p^{\text{next}}_i))\)
5: end for
6: return \(t_{\text{max}}\)

### 2.3.3 The GPM Second Stage

The second stage of the RRT-GPM algorithm formulates the multiple spacecraft reconfiguration maneuver as an optimal control problem. This optimal control problem is discretized at some specific discretization points called the Legendre-Gauss (LG) points, and then transcribed into a nonlinear program (NLP) by approximating the states and controls using Lagrange interpolating polynomials. The resulting NLP is then solved using the SNOPT nonlinear solver \[48\]. The augmented RRT output of the first stage is used as an initial guess in solving the NLP, and is essential in 1) reducing the computation times of the solution process (See Chapter 3) and 2) solving more complex reconfiguration problems (See Section 2.4).

Pseudospectral methods have been a popular choice among numerical direct methods to solve optimal control problem due to their ability to provide accurate solutions of the costates and other covectors, without requiring the use of analytically differential equations of the adjoints \[49\]. Another important feature of the pseudospectral methods is that they typically have faster convergence rate than other direct methods. They are known to demonstrate a “spectral accuracy” \[50\].

This thesis uses a Gauss pseudospectral method, which has been shown to have, in general, more accurate solutions than other pseudospectral methods \[39\]. Another
characteristic that distinguishes GPM among other pseudospectral methods is that the Karush-Kuhn-Tucker (KKT) conditions of the NLP have been shown to be exactly equivalent to the discretized form of the first-order optimality conditions of the Hamiltonian boundary value problem (HBVP) \[51\]. Therefore a solution to the NLP is guaranteed to satisfy the optimality conditions traditionally used in indirect methods, thus removing a primary disadvantage of direct methods. Ref. \[39\] and \[51\] are excellent references for pseudospectral methods, and more specifically for the Gauss pseudospectral method.

The general formulation adopted in this second stage is the following \[51, 52\]. Determine the state, \(x(t)\), and control, \(u(t)\), that minimize the cost functional

\[
J = \Phi(x(t_0), t_0, x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t)dt
\]

subject to the dynamic constraints

\[
\dot{x} = F(x(t), u(t), t) \in \mathbb{R}^n
\]

the boundary condition

\[
\phi(x(t_0), t_0, x(t_f), t_f) = 0 \in \mathbb{R}^q
\]

the inequality path constraints

\[
C(x(t), u(t), t) \leq 0 \in \mathbb{R}^c
\]

where \(t_0\) is the initial time, \(t_f\) is the final time, and \(t \in [t_0, t_f]\).

The optimal control problem of equations (2.25)-(2.28) is referred as the continuous Bolza problem. This problem is defined on \([t_0, t_f]\), where \(t_0\) and \(t_f\) can be free or fixed variables. However, the Gauss pseudospectral method used to solve this problem requires a fixed time interval, such as \([-1, 1]\). The mapping between the time
interval \( t \in [t_0, t_f] \) and the time interval \( \varsigma \in [-1, 1] \) can be written as

\[
\varsigma = \frac{2t}{t_f - t_0} - \frac{t_f + t_0}{t_f - t_0}
\]

(2.29)

Rewrite the optimal control problem after replacing \( t \) with \( \varsigma \):

\[
J = \Phi(x(-1), t_0, x(1), t_f) + \frac{t_f - t_0}{2} \int_{-1}^{1} g(x(\varsigma), u(\varsigma), \varsigma; t_0, t_f) d\varsigma
\]

(2.30)

subject to the constraints

\[
\dot{x} = \frac{t_f - t_0}{2} F(x(\varsigma), u(\varsigma), \varsigma; t_0, t_f) \in \mathbb{R}^n
\]

(2.31)

\[
\phi(x(-1), t_0, x(1), t_f) = 0 \in \mathbb{R}^q
\]

(2.32)

\[
C(x(\varsigma), u(\varsigma), \varsigma; t_0, t_f) \leq 0 \in \mathbb{R}^c
\]

(2.33)

where \( \varsigma \in [-1, 1] \). (2.30)-(2.33) is called the transformed continuous Bolza problem.

In the GPM, the set of \( N \) discretization points includes \( K = N - 2 \) interior LG points, the initial point \( \varsigma \equiv 0 \), and the final point \( \varsigma \equiv 1 \). GPM approximates the states by using a basis of \( K+1 \) Lagrange interpolating polynomials, \( L_i, i = 0 \ldots K \),

\[
x(\varsigma) \approx X(\varsigma) = \sum_{i=0}^{K} X(\varsigma_i) L_i(\varsigma)
\]

(2.34)

where

\[
L_i(\varsigma) = \prod_{j=0, j \neq i}^{K} \frac{\varsigma - \varsigma_i}{\varsigma_i - \varsigma_j}
\]

(2.35)

The control is approximated using a basis of \( K \) Lagrange interpolating polynomials \( L_i^\dagger, i = 1 \ldots K \)

\[
u(\varsigma) \approx U(\varsigma) = \sum_{i=1}^{K} U(\varsigma_i) L_i^\dagger(\varsigma)
\]

(2.36)
where
\[
\mathcal{L}_i^l(\varsigma) = \prod_{j=1, j \neq i}^{\kappa} \frac{\varsigma - \varsigma_j}{\varsigma_i - \varsigma_j} \quad (2.37)
\]

The dynamic constraints are transcribed into algebraic constraints as follows
\[
\sum_{i=0}^{\kappa} D_{ki} X_i - \frac{t_f - t_0}{2} F(X(\varsigma_k), U(\varsigma_k), \varsigma_k; t_0, t_f) = 0 \quad (2.38)
\]

where \(k = 1 \ldots \kappa\), and \(D\) is an \(\kappa \times (\kappa + 1)\) differential approximation matrix, consisting of the derivative of each Lagrange polynomial corresponding to the state at each LG point. This matrix can be computed offline as follows:
\[
D_{ki} = \dot{L}_i(\varsigma_k) = \sum_{l=0}^{\kappa} \frac{\prod_{j=0, j \neq l}^{\kappa} (\varsigma_k - \varsigma_j)}{\prod_{j=0, j \neq i}^{\kappa} (\varsigma_i - \varsigma_j)} \quad (2.39)
\]

where \(k = 1 \ldots \kappa\) and \(i = 0 \ldots \kappa\). Note that the collocation of the dynamic constraint only happens at the LG points and not at the boundary points. Two additional variables, \(X_0\) and \(X_f\) are defined in this discretization. \(X_0 \equiv X(-1)\), and \(X_f\) is defined via a Gauss quadrature
\[
X_f \equiv X_0 + \frac{t_f - t_0}{2} \sum_{k=1}^{\kappa} w_k F(X(\varsigma_k), U(\varsigma_k), \varsigma_k; t_0, t_f) \quad (2.40)
\]

where \(w_k\) are the Gauss weights.

Continuing with the transcription process, (2.30) is approximated using a Gauss quadrature
\[
J = \Phi(X(-1), t_0, X(1), t_f) + \frac{t_f - t_0}{2} \sum_{k=1}^{\kappa} w_k g(X(\varsigma_k), U(\varsigma_k), \varsigma_k; t_0, t_f) \quad (2.41)
\]

The boundary constraint is written as
\[
\phi(X(-1), t_0, X(1), t_f) = 0 \quad (2.42)
\]
Finally, the path constraint is computed at the LG points as

\[ C(X(\varsigma_k), U(\varsigma_k), \varsigma_k; t_0, t_f) \leq 0 \]  \hspace{1cm} (2.43)

where \( k = 1 \ldots K \). Equations (2.38), (2.40), (2.41), (2.42) and (2.43) form an NLP that is the transcription of the modified continuous Bolza problem (MCBP). The solution of the NLP is an approximate solution to the MCBP.

\section*{2.4 Examples}

In this section, examples of different complexity are solved using the RRT-GPM technique. In the figures illustrating the examples, the trajectories are shown in solid lines, and each dot represents a time step. The spacecraft are shown on the trajectory every sixth to tenth time step, depending on the example. The plot axes represent the axes of the local inertially fixed frame. The vectors attached on each spacecraft are the \( X \), \( Y \), and \( Z \) body axes. Some examples also include some fixed obstacles, shown as green sphere-shaped objects. Furthermore, examples that have “stay outside” constraints show red "umbrellas", with a handle showing the direction of the restricted pointing, and a cone of rays illustrating the angle of the constraint. Examples occasionally have circles around each spacecraft to explicitly show the boundaries of the spacecraft. The characteristics of the spacecraft are similar to those of SPHERES \[41\], and the dimensions of the test environment are similar to those of the SPHERES testbed on ISS. Note that some of the examples are motivated by reconfiguration maneuvers described in Ref. \[10\].

\subsection*{2.4.1 Implementation Details}

The RRT first stage and the transition step are programmed in C++, and they are compiled in Microsoft Visual Studio .NET 2003. The RRT first stage uses a linear solved based on the GLPK library. The GPM second stage is programmed in Matlab, and uses the GPOCS software package \[43\]. GPOCS is a MATLAB implementation.
of the Gauss pseudospectral method for solving optimal control problems. GPOCS relies on SNOPT \([48]\), an SQP solver for large-scale constrained optimization, to solve the NLP formed by the GPM method. The experiments are run on a Pentium 4, 2.2 GHz processor equipped with 1GB of RAM. The number of discretization points used in the second stage is \(N = 25\).

### 2.4.2 Example: Two-Spacecraft Maneuver with Obstacle and Sun Avoidance

The first example is a two-satellite maneuver that includes obstacle avoidance and sun avoidance constraint. Spacecraft 1 and 2 are initially positioned at \([0, 0, 0]^T\) and \([1, 1, 1]^T\). The maneuver consists of spacecraft 1 and 2 switching positions, and rotating 90° about the inertial Z-axis, while avoiding pointing at the sun and colliding with the a fixed obstacle. The unit vector pointing at the sun is the vector \([\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0]^T\), surrounded by a cone of 25° half angle. The spacecraft must maintain its “sensitive” instrument (e.g., telescope lens), which is mounted in the direction of the body X axis, out of the cone. The fixed obstacle is a sphere centered at \([0.3, 0.3, 0.3]^T\) with a radius of 0.15 m. The spacecraft are assumed to have a radius of 0.125 m. The computation times are 2 sec for the first stage, and 65 sec for the second stage. Figure 2-1 shows the output of first stage. The trajectory of both spacecraft are feasible, \(i.e.,\) the spacecraft avoid colliding with the fixed obstacle, and do not point in the sun cone. However, the trajectories are clearly suboptimal. Figure 2-2 shows the final trajectory, after the RRT output is smoothed through the second stage. The spacecraft follow a trajectory that satisfies all the constraints, and minimizes energy consumption.

Figure 2-3 and Figure 2-4 illustrate some of the constraints of this example over time. Figure 2-3 shows the obstacle avoidance constraint of both spacecraft. It displays the distance between the center of each satellite and the center of the obstacle. The plots show that these distances are at least larger than the minimum distance allowed between the spacecraft and the obstacle, which is shown as a dashed straight
Figure 2-1: Example: Two-Spacecraft Maneuver with Obstacle and Sun Avoidance. RRT Output. Spacecraft 1 and 2 switch positions while avoiding colliding with a fixed obstacle and pointing to the sun.

Figure 2-2: Final trajectory of the two-spacecraft maneuver with obstacle and sun avoidance example.
line. It is clear than the constraints are always feasible, and they are active in some parts of the trajectory. Figure 2-4 displays the sun avoidance constraints of both spacecraft, as the cosine of the angle between the sensitive instrument of the spacecraft and the absolute sun direction. It shows that the cosine of the angles is smaller than the maximum allowed value \textit{i.e.}, the sensitive instruments always stay outside the sun cone. So both figures show that constraints are satisfied during the whole maneuver.

2.4.3 Example: Coupled Two-Spacecraft Maneuver

This example is a more complex two-spacecraft reconfiguration maneuver. Its complexity is due to the inclusion of inter-spacecraft pointing constraints, which add coupling between the position and attitude states of the spacecraft. In this example, spacecraft 1 and 2 are initially positioned at \([0,0,0]^T\) and \([1,1,1]^T\). The maneuver consists of the spacecraft switching positions and rotating 180° about the inertial Z-axis, while satisfying an inter-spacecraft pointing constraint and avoiding collisions with the two fixed obstacles. Note that both spacecraft must point their body X axis (solid blue) to the other spacecraft to within 32°. The centers of the obstacles are located at \([0.5,0.7,0.5]^T\) and \([0.5,0.1,0.5]^T\), and have a radius of 0.15 m. The computation times are 4 sec for the first stage, and 112 sec for the second stage.

Figure 2-5 shows the trajectory output after the RRT stage is completed. Notice that both spacecraft maintain their relative pointing, and avoid colliding with the obstacles. Spacecraft 1 passes under the obstacle shown on the right of the figure, and Spacecraft 2 goes over the obstacle shown on the left. However, this strategy is clearly suboptimal, since there is enough space for both spacecraft to go in between the obstacles, and thus reduce fuel consumption. Figure 2-6 shows the final trajectory produced by the second stage of RRT-GPM path planner. As expected, the trajectory of both spacecraft were moved towards the diagonal path, while maintaining feasibility of the constraints.

This maneuver is designed to have a narrow passage in between the obstacles, which is usually hard for path planners to find. Figure 2-8 shows the inter-spacecraft
Figure 2-3: Distances between the center of the spacecraft and the center of the fixed obstacle. The dashed line shows the minimum permissible distance.

Figure 2-4: Cosine of the angles between the sensitive instrument of each spacecraft and the sun vector.
collision avoidance constraint which is active for a large part of the maneuver, when the spacecraft enter the narrow passage between the obstacles. Figures 2-7 and 2-9 show the distance between the centers of spacecraft and the centers of each obstacle. The dashed lines show the minimum permissible distances. The figures illustrate how the spacecraft have to maneuver around the obstacles while avoiding colliding with them. Figure 2-10 shows the inter-spacecraft pointing constraints. Spacecraft 1 and 2 must keep their instrument, which is mounted on their body X axis, pointing at the other spacecraft to within 32°. Figure 2-10 shows that this is a restrictive constraint since it forces the instrument to be pointing within a thin cone during the entire maneuver. The inter-spacecraft pointing constraints always stay feasible except around $t = 40$ s and $t = 80$ s, but the deviations are within the feasibility tolerance specified in the GPOCS solver, and are thus acceptable. Finally, it is important to note that the first RRT stage is essential in enabling a solution to coupled maneuver problems similar to this example. In fact, the GPOCS solver failed to solve this problem every time it was not initialized with the RRT guess of the first stage.
Figure 2-5: Example: Two-Spacecraft Maneuver with Inter-Spacecraft Pointing and Obstacle Avoidance. RRT Output. Spacecraft 1 and 2 switch positions while avoiding colliding with two fixed obstacles and keep pointing to each other within 32 degrees.

Figure 2-6: Final trajectory of the two-spacecraft maneuver with obstacle and inter-spacecraft pointing.
Figure 2-7: Distances between centers of the spacecraft and center of obstacle located at \([0.5, 0.7, 0.5]^T\) for the coupled two-spacecraft maneuver.

Figure 2-8: Distance between the centers of the spacecraft for the coupled two-spacecraft maneuver. The dashed line shows the minimum permissible distance.
Figure 2-9: Distances between centers of the spacecraft and center of obstacle located at $[0.5, 0.1, 0.5]^T$ for the coupled two-spacecraft maneuver.

Figure 2-10: Cosines of angles between ranging device vector and relative position of both spacecraft for the coupled two-spacecraft maneuver.
2.4.4 Example: Four-Spacecraft Maneuver

This example is a more challenging reconfiguration maneuver. It involves four spacecraft with absolute pointing constraint and several inter-spacecraft constraints. So it is a highly coupled maneuver. Spacecraft 1 and 2 switch their positions and attitude while pointing their body X axis to each other within $33^\circ$. Spacecraft 1 and 2 are initially located at $[0, 0.7, 0]^T$ and $[0, 0, 0]^T$. Two other spacecraft 3 and 4 are "health-monitoring" spacecraft 1 and 2. Both spacecraft 3 and 4 must end at their respective starting position, $[0, -0.5, 0]^T$ and $[0, 1.2, 0]^T$. They also have to keep pointing their body X axis at both spacecraft 1 and 2 to within $30^\circ$. All four spacecraft must also avoid pointing their X body axis in the sun direction. The sun cone is represented by the vector $[1, 0, 0]^T$ pointing at the sun and surrounded by a $20^\circ$ half angle cone. In this example, the radius of the spacecraft radius is 18.5 cm. The computation times are 17 sec for the first stage, and 302 sec for the second stage.

The trajectory produced by the RRT first stage is shown in Figure 2-11. Notice that spacecraft 1 and 2 have to leave the X-Y plane in order to keep pointing to each other and avoid pointing in the sun direction. Consequently, spacecraft 3 and 4 have to leave the X-Y plane in order to keep both spacecraft 1 and 2 inside their respective pointing cones. This shows a clear coupling between the positions and attitudes of the spacecraft which is due to the absolute pointing and relative pointing constraints. All spacecraft also avoid colliding with each other. Figure 2-12 shows the final trajectory produced by the RRT-GPM planner. It is a smoothed version of Figure 2-11. Figure 2-12 shows that all the constraints are met. Again, GPOCS failed to solve this problem when the RRT guess of the first stage was not given to it as an initial guess. This shows the importance of the first stage in enabling a solution for complex reconfiguration problems that include coupled constraints.
Figure 2-11: Example: Four-Spacecraft Maneuver with Inter-Spacecraft Pointing and Absolute Pointing. RRT Output. Spacecraft 1 and 2 switch positions while keep pointing to each other within 33°. Spacecraft 3 and 4 keep both spacecraft 1 and 2 in their specified cone of 30°.
Figure 2-12: Final trajectory of the four-spacecraft maneuver with inter-spacecraft and absolute pointing.
2.5 Implementation Issues

2.5.1 MRPs versus Quaternions

This section is a brief review of attitude representation of quaternions and MRP. Then it justifies the choice of MRP over quaternions and all other attitude representations as the preferred representation in the formulation of the reconfiguration maneuver problems.

The quaternion representation is given by

\[
q = \begin{bmatrix}
q_0 \\
q_1 \\
q_2 \\
q_3
\end{bmatrix}
\]  

with

\[
\begin{align*}
q_0 & = \hat{n} \sin \left( \frac{\theta}{2} \right), \\
q_3 & = \cos \left( \frac{\theta}{2} \right)
\end{align*}
\]

where \(\hat{n}\) represents the unit vector that corresponds to the axis of rotation, and \(\theta\) represents the angle of rotation. The four components of the quaternion vector satisfy a normalization constraint

\[
q^T q = q_0^2 q_{02} + q_3^2 = 1
\]

Quaternion parametrization has received a considerable use in the spacecraft attitude control systems area [44]. The main reason is that it does not possess any of the analytical issues of the smaller dimension representations. It also provides the easiest way to restore the orthogonality of a rotation matrix, in the event that it has been lost due to the accumulation of numerical errors. The kinematic equations of motion
in quaternion representation can be written as

\[ \dot{q} = \frac{1}{2} \Xi(q) \omega \]  

(2.47)

where

\[
\Xi(q) = \begin{bmatrix}
q_3 & -q_2 & q_1 \\
q_2 & q_3 & -q_0 \\
-q_1 & q_0 & q_3 \\
-q_0 & -q_1 & -q_2
\end{bmatrix}
\]  

(2.48)

So another property of the quaternion representation is the simplicity of its governing kinematical equation, which is both linear and regular.

The modified Rodrigues parameters (MRPs) are given by

\[ \sigma = \frac{q_{02}}{1 + q_3} = \hat{n} \tan \left( \frac{\theta}{4} \right) \]  

(2.49)

They are a minimal attitude parametrization, since they only require three components to describe attitudes. The kinematic equation of motion in MRP notation is given in (2.6). It has the unique property that, among all known three parameter descriptions of attitude, its Jacobian matrix \( R(\sigma) \) (2.9) has orthogonal rows and columns. But the MRPs suffer from the singular behavior \( \|\sigma\| \to \infty \) as \( \|\theta\| \to 2\pi \). This behavior is easily verified by examining equation (2.49). However, for \( \|\theta\| \to \pi \), which is the case for all rotations defined in the formulation of the reconfiguration maneuver problems of this thesis, the MRPs are very well behaved [53]. All rotations up to and including all possible \( \|\theta\| \to \pi \) displacements are within the unit sphere \( \|\sigma\| = 1 \), which is a regular and near linear region.

To compare the performance of the MRP representation against the quaternion representation, the two-spacecraft examples of Section 2.4 are run using the two different attitude representations. Example 1 is described in Section 2.4.2, and example 2 is described in Section 2.4.3. The same initial RRT guess and the same tolerances
Table 2.1: Comparison of computation times for problems with MRP versus same problems with quaternions

<table>
<thead>
<tr>
<th></th>
<th>Example 1</th>
<th>Example 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRP</td>
<td>65 sec</td>
<td>112 sec</td>
</tr>
<tr>
<td>Quaternions</td>
<td>127 sec</td>
<td>376 sec</td>
</tr>
</tbody>
</table>

are used in both cases. Table 2.1 shows the summary of computation times of the second MRP stage.

The results show that the experiments run using the MRP representation have considerably faster computation times. The difference is even more noticeable in example 2 which includes inter-spacecraft constraints. Recall that the solution of example 2 includes a very narrow passage. These results show that an adequate attitude representation is essential in solving such difficult problems. Several other reasons can also explain why examples formulated with MRPs outperform those with quaternions:

1. Even though the quaternion kinematic equations are linear in the quaternion variables, they are nonlinear when the angular velocities are also unknown variables, since these equations consist of the product of the quaternions and the angular velocities.

2. The MRP representation is composed of three components compared to four for the quaternions. Therefore it requires 25% less memory storage, which is an essential factor in decreasing the computation times.

3. Most importantly, the MRPs are free of the normalization equation (2.46) of the quaternion representation. The satisfaction of the normalization equation, which is an equality constraint, adds to the complexity of the solution process, and therefore increases the time needed by the solver to converge to a local optimum.
2.5.2 Performances versus Tolerances

To study the effects of the optimality and feasibility tolerances on the performance of the nonlinear solver of the GPM stage, a simple one-spacecraft reconfiguration maneuver is designed and tested with a range of values of tolerances. Let us first define the optimality and feasibility tolerances. The SNOPT solver considers a point as a satisfactory solution if it satisfies the first order optimality conditions, to within certain tolerances, of the NLP defined in the GPM stage. The tolerance parameter on the primal variables is the feasibility tolerance, and the one on the dual variables corresponds to the optimality tolerance \[^{54}\]. The reconfiguration consists of moving one spacecraft around a fixed obstacle while avoiding pointing to the sun. Since the two types of tolerances are correlated, 3D plots best summarize the effects of each of them on both the running time and the cost of the experiment. The experiment is run with and without an initial guess from the RRT stage. Note that the computation times of the with RRT case include the time needed to generate the RRT initial guess.

Figures 2-13 and 2-14 show the cost and computation time respectively, as a function of the feasibility and optimality tolerances of the nonlinear solver, for the RRT initialized case. Figure 2-13 demonstrates that the cost decreases when the feasibility tolerance decreases. But for values of feasibility tolerance smaller or equal to \(10^{-4}\), the cost is almost constant. The optimality tolerance does no affect the total cost significantly. A simple explanation is that the solver satisfies the optimality tolerance faster than the feasibility tolerance. Figure 2-14 shows that the computation time increases with a decrease in the feasibility tolerance. Computation time is not clearly influenced by the optimality tolerance. The same explanation given before applies in this case: the optimality tolerance is satisfied faster than the feasibility tolerance, therefore the factor influencing the computation time is the feasibility tolerance. Notice also that there is a jump of 15 seconds when the feasibility tolerance goes from \(10^{-4}\) to \(10^{-5}\). Considering simultaneously both Figure 2-13 and Figure 2-14, a logical choice is to set both tolerances to \(10^{-4}\).

Figure 2-15 and Figure 2-16 show the cost and computation time respectively, as
Figure 2-13: Variation of Cost with Feasibility and Optimality Tolerances for a problem with RRT initial guess.

Figure 2-14: Variation of Computation Time with Feasibility and Optimality Tolerances for a problem with RRT initial guess.
a function of the feasibility and optimality tolerances of the nonlinear solver, for the case where no initial guess is provided. Figure 2-15 shows that for values of feasibility tolerance larger than $10^{-3}$, the cost decreases with a decrease in optimality tolerance. But these levels of feasibility tolerance imply non-negligible infeasibilities. Thus, for this range of feasibility tolerance, the low values of the cost should be not be taken as a reference. For feasibility tolerance values smaller than $10^{-4}$, the cost remains constant, implying that the optimal tolerance has no effect on its value. Figure 2-16 shows that the computation time increases faster with an increase in feasibility tolerance compared to an increase in optimality tolerance. Notice that there is a jump of 30 seconds when the feasibility tolerance changes from $10^{-4}$ to $10^{-5}$. Looking simultaneously at both Figure 2-15 and Figure 2-16, it is clear that the cost is steady when the feasibility tolerance decreases from $10^{-4}$ to $10^{-5}$. So a reasonable choice is to pick $10^{-4}$ as the value for both tolerances.

Two interesting observations arise when comparing the test with the RRT guess and the test without it:

1. The best cost value is $1.78 \times 10^{-4}$ in both cases. In the RRT initialized case, it is only 0.7% better than the worse case cost. On the other hand, for the non-RRT case, the best case cost is 7% better than the worse case one. This shows that the RRT guess makes the solution less sensitive to tolerances.

2. In the RRT initialized example, the worst case computation time is 51 seconds, which is 47% better than the worst case time of the non-RRT example. This again demonstrates that the RRT initial guess is essential in making the solver converge faster.

This section compared the performance (cost and computation time) of the nonlinear solver as a function of the optimality and feasibility tolerances. The conclusion is that the choice of the right tolerances is essential in reducing computation times while getting enough precision in the solution cost. The effect of the tolerances on the computation time and cost scales with the complexity of the problem. Therefore, some effort has to be invested in picking the right values for the tolerances that
Figure 2-15: Variation of Cost with Feasibility and Optimality Tolerances for a problem without RRT initial guess.

Figure 2-16: Variation of Computation Time with Feasibility and Optimality Tolerances for a problem without RRT initial guess.
achieve the best tradeoff between cost and computation time.

2.5.3 Number of Nodes in GPM Stage

This section investigates the choice of number $N$ of discretization nodes in the GPM stage. The accuracy of the discretization increases with the number of nodes. In fact, pseudospectral methods, unlike other discretization methods which are based on uniform support points, do not suffer from the Runge phenomenon \cite{51}. It is the phenomenon seen when the approximation error near the boundaries increases as the number of discretization points increases. However, increasing the number of discretization nodes in the GPM methods leads to larger computation times. A larger $N$ means larger Hessian matrices in the NLP, more variables for every state and control, thus more memory storage and longer computation times. So the next question is how to find the smallest $N$ that achieves the desired precision. One popular technique called mesh refinement \cite{27}, starts with a coarse grid (i.e. low number of discretization nodes), and if necessary, refines the discretization, and then repeats the optimization steps. However, such approaches can be very time consuming, and therefore not feasible for online planning of reconfiguration maneuvers. Another drawback is that, even if the precision is met for a low $N$, there is no guarantee that the solution is feasible between the nodes. Pseudospectral methods only guarantees feasibility at the nodes, therefore a post-verification of the constraints is required, especially for problems with complex path constraints. This verification is required after every iteration of the mesh refinement technique, thus it is time consuming.

In the examples of this chapter, a value of $N$ equal to 25 gave the required precision with reasonable computation times. One pass of post-verification was done to make sure the constraints were satisfied between the nodes. Another way to evaluate the choice of number of nodes is to compute the error in the final control. This error is the difference between the real value computed analytically and the value produced by interpolating the controls over the discretization nodes. An analysis published in Ref. \cite{51} shows that, for a number $N$ larger than 20, the error in the final control fluctuates in a small region. The interpolation is done using three different methods
Figure 2-17: Error in the final control of an optimal control problem with path constraints as a function of the number of nodes using three different interpolation techniques [51].

(See Figure 2-17). These results support the choice of the value of $N$ in the examples of this chapter.

### 2.6 Summary

This chapter presented a two-stage RRT-GPM path planning technique to solve multi-spacecraft reconfiguration maneuvers. These maneuvers are challenging because they include nonlinear attitude dynamics, non-convex constraints, and coupling between the different spacecraft states in the constraints. The RRT-GPM technique is based on separating the problem into two stages: the first stage consists of solving a simplified path planning problem without differential constraints using bidirectional RRTs [22].
The output of the first stage is then augmented in a *transition* step with feasible dynamics. The result is a feasible initial solution that is given to a second stage to be improved. The second stage is formulated as an optimal control problem based on a Gauss pseudospectral method [39], and solved using the GPOCS software along with the SNOPT nonlinear solver. The examples demonstrated the validity of the approach. They showed that RRT-GPM can solve reconfiguration maneuver problems up to four spacecrafts including absolute and inter-spacecraft pointing constraints, and obstacle and collision avoidance constraints. The computation times obtained are also reasonable for online implementation. Finally, several implementation issues were discussed. Solving them is essential in reducing the computation times of the RRT-GPM approach.
Chapter 3

Importance of RRT Initialization in Pseudospectral Methods

This chapter discusses the importance of initializing GPM methods, and more generally pseudospectral methods, with an initial feasible guess. It reviews the different initialization techniques that have been used for optimal control problems. Then it introduces the RRT technique as a way to initialize pseudospectral methods. Finally, it shows the improvement that the RRT initialization brings to spacecraft reconfiguration problems, which are solved using a Gauss pseudospectral method.

3.1 Initialization of Pseudospectral Methods

Pseudospectral methods parameterize the states and controls of a continuous control problem using a basis of global polynomials, and then transcribes the discretized problem into an NLP. The resulting NLP is solved using off-the-shelf nonlinear solvers. Traditionally, nonlinear solvers require a feasible initial point for their algorithms to converge to a local optimal solution. If this point is not provided by the user, the algorithm solves an auxiliary problem to compute it. Once this point is found, the algorithm either guarantees the feasibility of its iterates using a barrier function that keeps the iterates inside the boundary of the feasible set (e.g., interior point methods [55]), or it checks at every iteration to make sure the current point is in the
feasible region, otherwise reestablishes feasibility (e.g., sequential quadratic programming methods [48]).

Therefore, generating a “good” guess to the solver is definitely an important step towards reducing the number of iterations, and consequently the computation time, required to solve the NLP. A main motivation to put effort in reducing computation time is that it enables real-time path planning onboard spacecraft, thus reducing future space mission costs and increasing mission quality [56]. It is therefore highly desirable to be able to autonomously solve a path planning problem based on optimal control in real-time. Simplified versions of optimal control problems (e.g., linearized problems) have been already solved in real-time control. However, the main challenge is to implement nonlinear programs in real-time, and get consistent and reliable solutions [51]. Pseudospectral methods have the potential to be solved in real-time because they provide high accurate solutions even with a relatively small number of discretization nodes i.e., a smaller problem size [51]. Algorithms that can determine feasible initial guesses for pseudospectral methods will help reduce the computation time of these methods, and therefore make them closer for real-time applications.

Note that generating a feasible initial guess to a highly constrained nonlinear problem can be as complicated as solving the nonlinear problem itself, so many researchers have suggested ways to compute good enough initial guesses. The closer the good guess to feasibility, the faster the solver will converge to the optimal solution. There have been different suggestions on how to generate a good guess for the nonlinear solver to solve the transcribed optimal control problem:

1. One common way is that the user develops a good guess for the control using common-engineering-sense [57]. Then the states are computed by using numerical integration. The guess (states and controls) will most likely not be feasible, in the sense that it will not satisfy the boundary conditions. Then looking at the resulting guess and using knowledge of the problem, the user can create a new guess for the control, and so on, until the guess is good enough. But this process can be time consuming, and it is not guaranteed to converge to a feasible answer.
2. Another method is known as the mesh refinement technique [27]. The idea is to start with a coarse grid \((i.e.,\) low number of discretization nodes) and use any guess \((e.g.,\) randomly chosen) as an initial starting point for the nonlinear solver. Then, if necessary, refine the discretization \((i.e.,\) increase the number of discretization nodes), and then repeat the optimization steps using the output of the previous step as the initial guess of the current step. But this approach can also be very time consuming, and therefore not feasible for online planning of reconfiguration maneuvers.

3. A third way of initializing the NLP is by using a “warm” start approach \([48, 58]\). A warm start uses the output of a similar version of the NLP as the initial guess for the actual problem. A similar version is either exactly the same problem run previously \((i.e.,\) offline compared to online), or a simplified version of the problem. Warm starts are widely used for active set solvers, but there are still many difficulties in applying them to interior point methods \([59]\). In addition, using a warm start technique based on an offline computation is not suitable to online implementation where the environment is affected by disturbances, and where previous solutions might not therefore be even feasible. If a warm start process is to be efficient, the algorithm must be chosen with care.

4. Something similar to a warm start is called homotopy methods that first solve a simpler version of the problem and then continuously modify the solution towards to the originally desired problem statement.

The next section introduces a warm start technique based on a randomized planner, that computes an feasible initial feasible guess for a class of problems based on pseudospectral methods. This guess is the solution of a simplified version of a path planning problem without differential constraints.
3.2 RRT Planner Initialization of Pseudospectral Methods

This section introduces a method based on Rapidly-exploring Random Trees (RRTs) to initialize pseudospectral methods. RRTs is a new class of randomized motion planning algorithms that was introduced in [33, 34]. It was originally developed for planning under differential constraints, but it has been applied mostly in ordinary motion planning. Refer to Section 1.3.2 for more details about RRTs.

The main idea is to use an efficient warm start approach to initialize the NLP resulting from the transcription of the continuous optimal control problem. This warm start is formulated as an RRT path planning problem, with no differential constraints. The output of the RRT planner is then augmented with feasible dynamics that are propagated from source to destination using an algorithm that ensures feasibility at each node. This process of augmenting the output of the randomized planner is called the transition phase in this thesis. The resulting output is a feasible initial guess to the complete optimal control problem.

The thesis uses an improved version of the bidirectional rapidly-exploring random trees (RRT) that is described in Ref. [22]. The transition phase is developed in this thesis in Section 2.3.2. The bi-directional RRT algorithm and the transition phase are adapted to solve the multiple spacecraft reconfiguration maneuver problem. The RRT algorithm and the transition phase can be adjusted to suit different optimal control problem specifics. This idea is best summarized in Figure 3-1.

3.3 Illustration of the RRT improvement

This section illustrates the improvement that the RRT initialization step brings to the solution of an optimal control problem transcribed into an NLP using a recently developed pseudospectral method method called the Gauss pseudospectral method (GPM) (Section 2.3.3). This method has shown promise both in the accuracy of the solution and post-optimality analysis of optimal control problems [40]. The problem
Figure 3-1: Steps Involved in Generating a Feasible Initial Guess to the Optimal Control Problem. This guess serves as a starting point to the NLP resulting from the transcription of the optimal control problem using Pseudospectral methods.

is a multispacecraft reconfiguration maneuver with path constraints that is described in Section 2.2. To underline the improvement of the RRT step, several reconfiguration maneuvers of increasing complexity are solved twice: 1) using the RRT step to find a feasible initial guess \textit{i.e.}, following the two-stage approach described in Section 2.3, and 2) using a “cold” start approach which leaves it to the nonlinear solver to find an initial starting guess. Refer to Section 2.4 for details related to the figures illustrating reconfiguration maneuvers examples.

The examples consist of reconfiguration maneuvers that include stay outside constraints (\textit{e.g.}, sun avoidance), inter-spacecraft collision avoidance constraints, and obstacle avoidance constraints. These are hard non-convex constraints. But the examples do not include inter-spacecraft constraints such as those found in the examples of Section 2.4.3 and Section 2.4.4. The reason is that, when the cold approach was tried on those two examples, the nonlinear solver experienced numerical difficulties and failed to converge to a solution. Thus the RRT initial guess is essential in such examples. Recall that inter-spacecraft constraints are coupling constraints \textit{i.e.}, they affect, in a coupled way, the position and attitudes of the spacecraft. Thus, the feasible set of problems having coupled constraints is usually a very small region, which explains the difficulties of the nonlinear solver. The examples below are solved using GPOCS, a software package based on the Gauss pseudospectral method [43]. Both
optimality and feasibility tolerances are set to $10^{-4}$. The number of discretization points is $N = 25$. For more details about the implementation, refer to Section 2.4.1. The computation times and costs of the following examples are summarized in Table 3.1 and Table 3.2.

### 3.3.1 Single Spacecraft Maneuver

This example is a simple translation from the position $[0, 0, 0]^T$ to $[1, 1, 1]^T$ with a 180° rotation around the Z axis. The spacecraft has to avoid pointing its X axis in the sun direction, which is represented by the vector $\left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right]^T$, surrounded by a cone of 30° half angle. It also has to avoid colliding with a fixed obstacle centered at $[0.6, 0.5, 0.5]^T$, with radius equal to 0.15 m. Figure 3-2(a) shows the initial guess produced by the RRT planner. Figure 3-2(b) shows the final smoothed trajectory produced by GPOCS when initialized with the RRT initial guess. Figure 3-2(c) displays the final trajectory for this problem when no initial guess is provided to the software.

### 3.3.2 Diagonally Crossing Maneuver

This problem consists of a three-spacecraft maneuver. Spacecraft 1 and 2 start at $[0, 0, 0]^T$ and $[1, 1, 1]^T$, the opposite corners of a cube of side 1. They must switch position and make a 90° rotation around the inertial Z axis. A third spacecraft, spacecraft 3 starts at $[1, 0, 0]^T$ and ends at $[0, 1, 1]^T$. It also performs at 90° rotation. The maneuver of spacecraft three crosses diagonally those of the other two spacecraft. All three spacecraft have to avoid a fixed obstacle located at $[0.3, 0.3, 0.3]^T$ with radius 0.2 m. The same sun avoidance constraint described in Section 3.3.1 applies in this problem. Figure 3-3 shows the RRT initial guess, the final trajectory of the problem that uses the RRT guess, and the final trajectory that does not use any initial guess. Two circles surrounding the spacecraft help visualize the boundaries of the spacecraft.
Figure 3-2: RRT Path and Final Trajectories for the Simple Single Spacecraft Example
Figure 3-3: RRT Path and Final Trajectories for the Three-Spacecraft Example

(a) RRT Initial Guess

(b) Final Trajectory solved using RRT initial guess.

(c) Final Trajectory solved without using any initial guess.
3.3.3 Formation Reflection with Four Spacecraft

This example consists of four spacecraft that start in a square formation, and end in another reflected square formation. Spacecraft 1 and spacecraft 3 start at $[0, 1, 0]^T$ and $[2, 1, 0]^T$, respectively. They must end at their original positions. They must also rotate 90° around the inertial Z axis. Spacecraft 2 and 4 start at $[1, 0, 0]^T$ and $[1, 2, 0]^T$, and must switch their position. They must also rotate 180° around the inertial Z axis. All spacecraft must avoid pointing their X body axis inside two cones of 50° along the X and -X inertial directions. They must also avoid colliding with a fixed obstacle located at the center of the square, with radius 0.25 m. The pointing constraints lead to a non-trivial rotation maneuvers for both spacecraft 1 and 3. The fixed obstacle makes the trajectories of spacecraft 2 and 4 a more challenging one. In Summary, the maneuver is a reflection of a square formation around a line passing through the fixed positions of Spacecraft 1 and 3.

3.3.4 Formation Rotation with Five Spacecraft

This example consists of a five spacecraft starting in a pyramid formation. Spacecraft 5 starts at the apex of the pyramid, while spacecraft 1 to 4 form its square base. Each spacecraft must move to the next spacecraft position in the sequence (spacecraft 2 moves to spacecraft 1 position, 3 to 2, 4 to 3, 5 to 4, and 1 to 5). Each spacecraft must also end the maneuver pointing in the direction where the next spacecraft in the sequence was pointing at the beginning of the maneuver. Thus, the final configuration is a rotated version of the original pyramid formation. The spacecraft must avoid pointing their X body axis towards the sun direction represented by the vector $\left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right]^T$, surrounded by a cone of 20° half angle. They must also avoid colliding with two fixed obstacles of radius 0.15 m. Figure 3-4 shows the RRT initial guess, the final trajectory of the problem that uses the RRT guess, and the final trajectory that does not use any initial guess.
Figure 3-4: RRT Path and Final Trajectories for the Five-Spacecraft Example
Table 3.1: Comparison of computation times of reconfiguration maneuvers for formation of increasing size solved using a Gauss pseudospectral method (average over 10 runs)

<table>
<thead>
<tr>
<th>Example</th>
<th>Time (s) w/o RRT</th>
<th>Time (s) with RRT</th>
<th>Time Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 s/c (3.3.1)</td>
<td>35</td>
<td>24</td>
<td>1.46</td>
</tr>
<tr>
<td>2 s/c (2.4.2)</td>
<td>103</td>
<td>67</td>
<td>1.54</td>
</tr>
<tr>
<td>3 s/c (3.3.2)</td>
<td>335</td>
<td>171</td>
<td>1.96</td>
</tr>
<tr>
<td>4 s/c (3.3.3)</td>
<td>478</td>
<td>228</td>
<td>2.10</td>
</tr>
<tr>
<td>5 s/c (3.3.4)</td>
<td>834</td>
<td>356</td>
<td>2.34</td>
</tr>
</tbody>
</table>

### 3.4 Performance Comparison

Table 3.1 summarizes the computation times of the examples used to show the improvement of the RRT initialization on problems solved using pseudospectral methods. The *with RRT* time includes the time of the RRT step, the transition step, and the time required by the GPOCS to solve the problem. The *w/o RRT* consists of the time needed by GPOCS to solve the same problems without an initial RRT guess *i.e.*, using a cold start approach. The last column shows the ratio of the *w/o RRT* times over the *with RRT* times, *i.e.*, the computation time improvement due to the RRT initial guess.

Figure 3-5 displays side to side the computation times of both versions of the solutions of the reconfiguration maneuvers as a function of the formation size. The results show that the RRT initialization reduces the GPOCS computation time, when compared to the *w/o RRT* case, by a factor increasing from 1.46 to 2.34 with the size of the formation. Figure 3-6 illustrates this increase in time ratio. The time ratio values can be fit by a straight line with equation

\[
TimeRatio(N) = 0.23N + 1.18
\]  

where \( N \) is the size of the formation. Thus, the time ratio of using the RRT initial guess is approximately linear for these formation sizes.

It is also interesting to compare the computation scaling of each of the two cases.
Figure 3-5: Comparison of computation times of a series of reconfiguration maneuvers of increasing size solved using a Gauss pseudospectral method (average over 10 runs). The with RRT includes computation times of the RRT planner and the transition step.

By linearly fitting the logarithms of the ratio time values versus the logarithms of the size of the formation, it can be computed that

\[
\text{without RRT Time}(N) \approx 31.4N^{2.01} \quad (3.2)
\]
\[
\text{with RRT Time}(N) \approx 23.3N^{1.69} \quad (3.3)
\]

Thus, the w/o RRT computation time is approximately quadratic with the size of the formation, while the with RRT computation time is faster than linear, but slower than quadratic. Clearly, the RRT initial guess improves the scaling of the GPOCS computation time of the solution of the multiple spacecraft reconfiguration problem.

Table 3.2 summarizes the final costs of the same examples. The last column compares the cost of each example solved without an initial RRT guess to the cost of the same example solved with an RRT initial guess. A positive value indicates the
Figure 3-6: Time Ratios of \textit{w/o} RRT over \textit{with} RRT Computation Times. The \textit{with} RRT includes computation times of the RRT planner and the transition step.

percentage of improvement (\textit{i.e.}, decrease) in the cost of the \textit{w/o} RRT case when compared to the \textit{with} RRT one. One would expect that the \textit{w/o} RRT costs to be better than those of the \textit{with} RRT case. The reason is that, without an initial guess, the solver generates its own initial feasible guess that gives, on every run, the same best result it can find. The RRT guess, however, is based on a randomized planner, and therefore, can restrict the solver to solve the problem in a specific region, which might not be the best one. But, it can be seen from the cost results, that in average, the RRT guess did not have any noticeable impact on the final costs of the maneuvers. In the worst case, the cost of \textit{w/o} RRT version is 3.65\% better. For the largest formation, the five spacecraft reconfiguration described in Section 3.3.4, the \textit{w/o} RRT cost is only 1.02\% better.
Table 3.2: Comparison of final costs of reconfiguration maneuvers for formation of increasing size solved using a Gauss pseudospectral method (average over 10 runs)

<table>
<thead>
<tr>
<th>Example</th>
<th>Cost w/o RRT</th>
<th>Cost with RRT</th>
<th>Comparison %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 s/c (3.3.1)</td>
<td>0.975</td>
<td>1.012</td>
<td>+3.65</td>
</tr>
<tr>
<td>2 s/c (2.4.2)</td>
<td>1.806</td>
<td>1.847</td>
<td>+2.22</td>
</tr>
<tr>
<td>3 s/c (3.3.2)</td>
<td>2.752</td>
<td>2.821</td>
<td>+2.45</td>
</tr>
<tr>
<td>4 s/c (3.3.3)</td>
<td>4.490</td>
<td>4.444</td>
<td>−1.03</td>
</tr>
<tr>
<td>5 s/c (3.3.4)</td>
<td>8.654</td>
<td>8.743</td>
<td>+1.02</td>
</tr>
</tbody>
</table>

### 3.5 Summary

Pseudospectral methods, a relatively new class of direct methods, has become very popular as a solution technique to solve optimal control problems, mainly because of its ability to determine high accurate solutions of the costate vectors. However, one negative aspect of pseudospectral methods in general is that the computation time increases dramatically with the complexity of the problem. This chapter introduced an initialization technique based on RRTs to compute an initial feasible guess to the problem. This guess is responsible in 1) reducing the computation times of the solution, and 2) making it possible to solve more complex reconfiguration maneuvers. The examples of this chapter show that the RRT initial guess considerably reduces computational scalability of the solution time without a noticeable impact on the total cost. These examples include obstacle avoidance and absolute pointing constraints. They are solved using GPOCS, a software package based on a gauss pseudospectral method. It is also important to emphasize that more complex maneuvers including inter-spacecraft constraints are only solvable when using the RRT initialization technique. GPOCS fails to solve these examples when it is not supplied with a feasible initial guess. These examples are described in Section 2.4.3 and Section 2.4.4.
Chapter 4

Comparison with Other Solution Approaches

The experiments in Chapter 2 and Chapter 3 showed that two-stage RRT-GPM algorithm introduced in this thesis found solutions for complex reconfiguration maneuver problems with up to five spacecraft. In this chapter, the RRT-GPM approach is compared to two other two-stage trajectory planning approaches: RRT-Smooth and RRT-LPM. All three approaches use the Rapidly-exploring Random Tree (RRT) planner described in Section 2.3.1 as their first stage. The RRT-Smooth algorithm was introduced in Ref. [10]. Its second stage is based on an optimization that is solved iteratively using a linearization of the problem. We will refer to RRT-Smooth as the RRT-LS approach, where LS refers to “linear smoother” to avoid any ambiguity in the use of the term “smoother”. Our RRT-GPM approach has been motivated by the work done in [10]. It mainly improves the second stage by using a Gauss pseudospectral method. The RRT-LPM approach, where LPM refers to the Legendre pseudospectral method [37], is another two-stage technique developed in this thesis to solve the reconfiguration maneuver problem. It relies on the software package DIDO [57] to solve its second stage. DIDO is written in MATLAB, and is based on the LPM in the discretization of the optimal control problem. RRT-GPM and RRT-LPM are very similar methods theoretically, but they exhibit different performances.
4.1 RRT-LS versus RRT-GPM

4.1.1 The RRT-LS Approach

The full details of the RRT-LS approach can be found in [10, 21]. As mentioned earlier, the first stage of the RRT-LS approach is identical to the one described in Section 2.3.1. The second stage is based on iteratively solving a linear program resulting from the linearization of the cost function, dynamics, and constraints about the initial feasible solution. It is summarized in the following.

First, the complete trajectory is represented by the sequence of points \( p(k) \), \( k \in 0 \ldots \lceil T/\Delta T \rceil \), where \( \Delta T \) is the time step. This method assumes constant input forces and torques over each time step. These inputs propagate the point \( p(k) \) to the next point by using a propagation function \( f \)

\[
p(k + 1) \approx f(p(k)) \quad \forall k \in 0 \ldots \lceil T/\Delta T \rceil
\]

Thus (4.1) represents the discrete dynamics of the spacecraft. The pointing, obstacle avoidance, and collision avoidance constraints can be discretized as

\[
g_m(k) \leq 0, \quad k \in 0 \ldots \lceil T/\Delta T \rceil
\]

for every constraint \( m \).

An iteration of the second stage of the RRT-LS algorithm therefore consists of computing a perturbation of the trajectory that improves the cost and maintains the feasibility of the trajectory at the same time. A first-order Taylor approximation is used in the update process,

\[
p(k + 1) + dp(k + 1) \approx f(p(k) + dp(k + 1)) \\
\approx p(k + 1) - f(p(k) + dp(k + 1) - \nabla f(p(k))^T dp(k) = 0
\]

(4.3)
and

\[ g_m(p(k) + dp(k)) \approx g_m(p(k)) + \nabla g_m(p(k))^T dp(k) \leq 0 \quad (4.4) \]

where \(\|dp(k)\| \leq \epsilon \ll 1\).

The discretized form of the cost can be written as

\[ J = \Delta T \sum_{k=0}^{\lfloor T/\Delta T \rfloor} \sum_{i=1}^{N} |f_i(k) + df_i(k)| + |\tau_i(k) + d\tau_i(k)| \quad (4.5) \]

which can be rewritten to make it suitable for linear programming as

\[ J = \Delta T \sum_{k=0}^{\lfloor T/\Delta T \rfloor} \sum_{i=1}^{N} a_i(k) + b_i(k) \quad (4.6) \]

subject to

\[ |f_i(k) + df_i(k)| \leq a_i(k), \quad \forall i, k \quad (4.7) \]
\[ |\tau_i(k) + d\tau_i(k)| \leq b_i(k), \quad \forall i, k \quad (4.8) \]

The second stage of the RRT-LS approach is summarized in Algorithm 4.1 and Algorithm 4.2. The updated solution in step 3 of Algorithm 4.2 may violate the constraints by a small amount. But the solution is repaired in Repair-Consistency using the inverse of the discretized dynamics equation in order to regain consistency. Repair-Consistency recomputes the velocities, forces and torques using the values of the positions and attitudes of the current iteration, therefore recovering the consistency between the states, controls and dynamics of the problem. The inconsistencies in the inequality constraints are not explicitly repaired. Therefore to make sure the
Algorithm 4.2 Linear-Smoother-Step\((p)\)

1: \textbf{for} \(j \leftarrow 1\) to \(N\) \textbf{do}
2: \hspace{1em} Solve linear program:
   \[
   \min_{\Delta T} \sum_{k=0}^{[T/\Delta T]} \sum_{i=1}^{N} a_i(k) + b_i(k) \quad \forall k
   \]
   subject to
   \[
   \begin{bmatrix}
   I & \nabla f(p_i(k))^T
   \end{bmatrix}
   \begin{bmatrix}
   dp_i(k + 1) \\
   dp_i(k)
   \end{bmatrix} = 0
   \]
   \[
   g_m(p(k)) + \nabla g_m(p(k))^T dp(k) \leq 0
   \]
   \[
   |f_i(k) + df_i(k)| \leq a_i(k)
   \]
   \[
   |\tau_i(k) + d\tau_i(k)| \leq b_i(k)
   \]
   \[
   |dp_i(k)| \leq \epsilon
   \]
   \hspace{1em} \textbf{End of linear program}
3: \(p_i(k) \leftarrow p_i(k) + dp_i(k), \quad \forall k\)
4: \textbf{Repair-Consistency}(\(p_i(k)\)), \quad \forall k
5: \textbf{end for}
6: \textbf{return} \(p\)

final answer is feasible, the constraints bounds are increased by a small margin before the start of the algorithm.

4.1.2 Comparison Example 1

To compare the accuracy and performance of RRT-LS with RRT-GPM, a similar example to the one described in Section 3.3.1 is solved again using the RRT-GPM and the RRT-LS approaches. Recall the example of Section 3.3.1 is a simple maneuver of one spacecraft, consisting of a translation from one corner of a cube to the opposite corner, and a rotation around the Z body axis. The problem also contains sun avoidance and obstacle avoidance constraints. In this example, the obstacle is located at \([0.3, 0.3, 0.3]^T\) with radius 0.1 m, and there are two \(20^\circ\) “stay outside” pointing constraints in the +Y and −Y inertial axis directions. The spacecraft characteristics are those of SPHERES [41]. They are given in Section 5.1. This example was also performed onboard the International Space Station (ISS), and the theoretical results will be compared to the flight results in Chapter 5.

Even though this problem might appear to be a simple maneuver, it does not have an analytical solution. But the Gauss pseudospectral method, which is used in the RRT-GPM approach, has been shown to find the optimal solutions to such problems.
In fact, these solutions have been shown to satisfy the Karush-Kuhn-Tucker (KKT) optimality conditions of the NLP [39]. Furthermore, the KKT conditions of the NLP have been proven to be exactly equivalent to the discretized form of the continuous first-order necessary conditions of the Bolza problem (refer to Section 2.3.3) when using the Gauss pseudospectral discretization [51]. Therefore, the solution returned by the RRT-GPM is indeed locally optimal.

This comparison uses quaternions instead of MRPs to represent attitude of the spacecraft. The reason is that, in the original RRT-LS approach, quaternions were used in the attitude representation. But this choice does not affect the feasibility and optimality of the solution. The cost expression in the RRT-LS approach (4.6) is also modified in this comparison. The absolute values are changed to the square of Euclidean norms. The reason is that the absolute value is a non-smooth function that presents additional difficulties to the non-linear solver the in RRT-GPM approach. But recall that RRT-LS is based on linearizing the cost function along with the constraints. Thus, the cost function used by the RRT-LS technique is a linearization of the quadratic cost of the RRT-GPM approach (2.41). Finally, the length of the maneuver is designed to be 66 seconds in both tests.

Figure 4-1 shows the final trajectories using the two different approaches. The trajectories look similar. In both trajectories, the spacecraft avoids colliding with the fixed obstacle, and keeps its X body axis out of the restricted cones. To compare these trajectories further, the states and controls of the solutions are plotted over time (See Figures 4-2 and 4-4). As expected, the states computed by the RRT-LS approach have a more linear behavior than the RRT-GPM states. However, the RRT-GPM states have “smoother” plots. Figure 4-3 and Figure 4-5 show the optimal controls over time computed by the two approaches. These figures show that the two approaches have very different optimal controls (forces and torques): in the RRT-GPM approach, the controls are distributed over the time of the maneuver, while in the RRT-LS, the controls are zeros (or very close to zero) except at few periods of time where they have an impulse-like behavior. This behavior is due to the linear programming solution method embedded in the RRT-LS approach. In fact, RRT-
LS uses the GLPK software package, and chooses simplex as the solution method. The impulse-like behavior is directly related to the basic/non-basic variables in the optimal solution of the simplex method [60].

The final trajectory produced by the RRT-LS approach has a cost of 0.0126 and is returned after 27 seconds. Comparatively, the trajectory returned by the RRT-GPM approach has a cost of 0.00179, and it is produced after 63 seconds. The results show that, even though the RRT-GPM approach is slower than the RRT-LS approach, it returns a considerably more optimal solution. In this example, the cost of final trajectory is almost 7 times smaller in the RRT-GPM case.

4.1.3 Comparison Example 2

This example consists of a two-spacecraft reconfiguration maneuver involving coupled constraints. It is similar to the example described earlier in Section 2.4.3. This example was also performed abroad the International Space Station (ISS). The flight results will be compared to the theoretical results in Chapter 5. The spacecraft have to switch positions while pointing their X body axis at each other to within 35°. They also have to avoid colliding with a fixed obstacle of radius 0.12 m, located in between their initial positions. The cost functions used in the design of the maneuvers are the same as those of Section 4.1.2. Finally, the length of the maneuver is designed to be 47 seconds in both versions of the test.

Figure 4-6 shows the final trajectories using the two different approaches. In both trajectories, the spacecraft satisfy all the path constraints. In Figure 4-7 and Figure 4-8, the controls of each spacecraft are plotted over time. Similar to the last example, the controls of the RRT-LS approach have an "impulse-like" behavior for both spacecraft. The same explanation applies here. In the RRT-GPM approach solution, the controls are more continuous, but they are very nonlinear. This is due to the nonlinearity of the attitude dynamics equations of the spacecraft.

The final trajectory produced by the RRT-LS approach has a cost of 0.0349 and is returned after 52 seconds. Comparatively, the trajectory returned by the RRT-GPM approach has a cost of 0.00738, and it is produced after 315 seconds. The slow time
Figure 4-1: Final Trajectories for the single spacecraft maneuver.

(a) RRT-GPM Approach

(b) RRT-LS Approach
Figure 4-2: States of the spacecraft over time in the solution of the single spacecraft problem using the RRT-GPM approach.

Figure 4-3: Controls of the spacecraft over time in the solution of the single spacecraft problem using the RRT-GPM approach.
Figure 4-4: States of the spacecraft over time in the solution of the single spacecraft using the RRT-LS approach.

Figure 4-5: Controls of the spacecraft over time in the solution of the single spacecraft using the RRT-LS approach.
of the RRT-GPM approach is mostly due to the use of quaternions in the attitude representation. The same example run using the RRT-GPM approach with MRPs instead of quaternions returns the optimal solution with the same optimal cost in 143 seconds. Nevertheless, this example confirms the results of the previous one: the RRT-GPM is slower than the RRT-LS approach, but its solution cost is considerably better. In this example, the cost of the final trajectory is almost 5 times smaller in the RRT-GPM approach.

4.1.4 Discussion

Below are some observations related to the results obtained in Section 4.1.2 and Section 4.1.3:

- While the RRT-GPM approach relies on the SNOPT nonlinear solver to compute the optimal solution in its second state, the RRT-LS approach is based on the GLPK linear solver. As shown in Figure 4-3, Figure 4-5, Figure 4-7 and Figure 4-8, the optimal controls computed by the two different approaches are very different. In the RRT-LS approach, the constraints and the cost are linearized around an initial trajectory. The linearization is prone to errors, especially when it is performed on nonlinear functions as the attitude dynamics of the spacecraft or the normalization equation of the quaternions. On the other hand, the second stage of RRT-GPM approach has been shown to return solutions that satisfy the KKT conditions of the NLP. These solutions are therefore locally optimal, and usually converge to global optimal solutions when provided with a good initial guess.

- The RRT-GPM approach computes trajectories with considerably lower costs when compared to the RRT-LS approach. The major reason is that the reconfiguration maneuver is a nonlinear problem, and posing it as a linear problem does not optimally solve the original problem. However, the computation times of the RRT-LS approach is considerably lower than the RRT-GPM approach, even after taking into account that computation times of the RRT-GPM approach
Figure 4-6: Final Trajectories for the two-spacecraft maneuver.
Figure 4-7: Controls of the spacecraft over time in the solution of the two-spacecraft using the RRT-GPM approach.
Figure 4-8: Controls of the spacecraft over time in the solution of the two-spacecraft using the RRT-LS approach.
are improved when the quaternions are replaced by MRPs in the attitude representation. Therefore, there is a tradeoff between the cost and the computation time when comparing the two approaches. Nevertheless, the RRT-GPM approach has a real advantage over the RRT-LS approach because of the accuracy and optimality of its solutions.

- There is a hidden burden in the RRT-LS approach implementation. In Algorithm 4.2, the improvement on the current point \( p \) is limited to be smaller than a specified vector \( \epsilon \). The comparison is understood to be component-wise. The difficulty lies in the choice of the values of the components of the \( \epsilon \) vector. These values are sensitive to the size of the problem \( i.e., \) number of spacecraft, the type of constraints, and the dimensions of the problem environment. This choice is a time consuming trial and error technique. It is not needed in the RRT-GPM approach.

### 4.2 RRT-LPM versus RRT-GPM

#### 4.2.1 The RRT-LPM Approach

The RRT-LPM and RRT-GPM approaches are very similar. They both use the same RRT planner described in Section 2.3.1 in their first stage, and a pseudospectral method in their second stage. The main difference is that RRT-LPM uses a Legendre pseudospectral method, while RRT-GPM employs a Gauss pseudospectral method. In addition, the RRT-LPM approach relies on the software package DIDO \[57\] to solve the NLP in the second stage of its two-stage path planning approach. DIDO utilizes a Legendre Pseudospectral method (LPM) as the technique to discretize the continuous optimal control problem into the NLP.

In the LPM, the support points used to approximate the states and controls are called the Legendre-Gauss-Lobatto (LGL) points. The main difference between the LGL points and the LG points used by the GPM method is that the LGL points include the boundaries points and interior points while the LG points only include...
interior points. The interest reader is encouraged to consult Ref. [51] for more details about different pseudospectral methods, and an elaborated comparison between them. The LPM formulation is provided next for completeness.

In the LPM, the set of discretization points and collocation LGL points are equivalent. Let us assume there are \( K \) collocation points. LPM approximates the states by using a basis of \( K \) Lagrange interpolating polynomials, \( \mathcal{L}_i, i = 1 \ldots K, \)

\[
x(\varsigma) \approx X(\varsigma) = \sum_{i=1}^{K} X(\varsigma_i) \mathcal{L}_i(\varsigma) \tag{4.9}
\]

The control is defined similarly to the state

\[
u(\varsigma) \approx U(\varsigma) = \sum_{i=1}^{K} U(\varsigma_i) \mathcal{L}_i(\varsigma) \tag{4.10}
\]

The dynamic constraints are transcribed into algebraic constraints as follows

\[
\sum_{i=1}^{K} D_{ki} X_i - \frac{t_f - t_0}{2} F(X(\varsigma_k), U(\varsigma_k)) = 0 \tag{4.11}
\]

where \( k = 1 \ldots K \), and \( D \) is an \( K \times K \) differential approximation matrix, consisting of the derivative of each Lagrange polynomial corresponding to the state at each LGL point. Note again that, unlike in the GPM, the collocation of the dynamic constraint happens at all discretization points which include the boundary points.

Continuing with the transcription process, the cost function is approximated as follows

\[
J = \Phi(X(-1), X(1)) + \frac{t_f - t_0}{2} \sum_{k=1}^{K} w_k g(X(\varsigma_k), U(\varsigma_k)) \tag{4.12}
\]

where \( w_k \) are the Legendre weights.

\[
\phi(X(-1), X(1)) = 0 \tag{4.13}
\]
Finally, the path constraint is computed at the LGL points as

\[ C(X(\varsigma_k), U(\varsigma_k)) \leq 0 \]  \hspace{1cm} (4.14)

where \( k = 1 \ldots K \). Equations (4.11), (4.12), (4.13) and (4.14) form an NLP that is the transcription of the modified continuous Bolza problem (MCBP). The solution of the NLP is an approximate solution to the MCBP.

Next are two examples that compare the performance of RRT-LPM against RRT-GPM. These examples are two maneuvers that have been presented in the previous section to compare RRT-LS against RRT-GPM. But note that in this section, MRPs are used in the attitude representation. The reason is that better computation times are achieved with MRPs compared to quaternions (See Section 2.5.1). The optimal states and costs computed by the RRT-GPM are not affected by the change of attitude representation. Therefore, only the states and controls of the RRT-LPM approach will be displayed in the following examples.

### 4.2.2 Comparison Example 1

First, the same example described in Section 4.1.2 is solved again using the two approaches: RRT-LPM and RRT-GPM. Recall that this example is a single spacecraft maneuver with “stay outside” and obstacle avoidance constraints.

Figure 4-9 shows the final trajectory returned by the RRT-LPM approach. It looks almost identical to Figure 4-1(a), the final trajectory computed by the RRT-GPM approach. To compare the two approaches further, the states and the controls of the RRT-LPM solution are plotted over time in Figure 4-2 and Figure 4-3. Note that in Figure 4-10, the attitude representation is plotted in quaternions instead of MRPs for easier comparison with the Figure 4-2. Comparing Figure 4-10 and Figure 4-2 show that the final states of the spacecraft have a very similar behavior. The same observation applies for the controls displayed in Figure 4-11 when compared to Figure 4-3: the two approaches return similar controls. These observations are expected because the RRT-GPM and RRT-LPM are both based on the same type of
Figure 4-9: Final Trajectories for the single spacecraft maneuver using the RRT-LPM approach.

The final trajectory produced by the RRT-LPM approach has a cost of 0.00213. Comparatively, the trajectory returned by the RRT-GPM approach has a cost of 0.00179, which is 16% better. But the main difference is in the computation times. The RRT-GPM approach solves this example in 26 seconds, while the RRT-LPM requires 253 seconds to complete, almost 10 times longer.

4.2.3 Comparison Example 2

This example has been introduced in Section 4.1.3. It is solved again here using the two approaches: RRT-LPM and RRT-GPM. Recall that this example consists of a two-spacecraft maneuver that includes inter-spacecraft pointing and obstacle avoidance constraints.

Figure 4-12 shows the final trajectory computed by the RRT-LPM approach. It has similarities with Figure 4-6(a). Figure 4-13 displays the optimal controls for both spacecraft. Comparing Figure 4-13 and Figure 4-7 shows that the two approaches
Figure 4-10: States of the spacecraft over time in the solution of the single spacecraft using the RRT-LPM approach.

Figure 4-11: Controls of the spacecraft over time in the solution of the single spacecraft using the RRT-LPM approach.
return different controls, but the general behavior is similar. The forces are almost linear, while the torques are clearly nonlinear. This can be explained by the linearity of the translation dynamics and the nonlinearity of the attitude dynamics.

The final trajectory produced by the RRT-LPM approach has a cost of 0.00986. Comparatively, the trajectory returned by the RRT-GPM approach has a cost of 0.00738, which is 25% better. But again, the main difference is in the computation times. The RRT-GPM approach solves this example in 143 seconds, while the RRT-LPM requires 1135 seconds to complete, almost 8 times longer.

### 4.2.4 Discussion

RRT-GPM and RRT-LPM are both based on pseudospectral methods, and use the nonlinear solver SNOPT [48] as part of the second stage of their solution approach. The main difference is in their implementation. The software package DIDO [57] is used in the RRT-LPM approach, the software package GPOCS [43] in the RRT-GPM
Figure 4-13: Controls of the spacecraft over time in the solution of the two-spacecraft using the RRT-LPM approach.

approach. GPOCS offers two features that DIDO does not have: 1) control of the optimality and feasibility tolerances, and 2) auto-scaling feature. First, as shown in Section 2.5.2, the choice of tolerances is crucial in reducing the computation times while maintaining the same level of optimality in the solution. DIDO uses the default tolerances which are set to $10^{-6}$, and there is no access to change their values. Next, the auto-scaling feature that GPOCS offers also helps in improving the computation times. It is the responsibility of the designer to scale its problem well, but having an auto-scale feature guarantees that the problem given to the nonlinear solver is well posed. Thus the auto-scale feature improves the rate of convergence of the nonlinear solver. The examples also show that the RRT-GPM approach achieves a better
optimal cost compared to the RRT-LPM approach. There is no clear explanation for this improvement. One possible reason is that these two approaches are based on two different pseudospectral methods that discretize the optimal control problem on two different set of discretization points. In addition, it has been shown that the error in the states and controls, when compared to the true solution, is smaller in the GPM compared to the LPM [51]. As a conclusion, the examples demonstrate that RRT-GPM approach outperforms the RRT-LPM approach in both computation time and final cost.

4.3 Summary

This chapter compares the RRT-GPM approach against two other approaches: RRT-LS and RRT-LPM. Even though RRT-GPM has worse computation time than RRT-LS, its approach is considerably more accurate and achieve far better final costs. The main reason is due to the linearization errors that the RRT-LS introduces in the solution. RRT-GPM has been next shown to outperform the RRT-LPM approach both in the computation times and in the optimality of the solution. The computation times of RRT-LPM are considerably longer mainly because of the lack of control over the tolerances of the nonlinear solver and the absence of auto-scaling in the DIDO software.
Chapter 5

Experiments on SPHERES onboard ISS

This chapter describes several reconfiguration maneuver experiments performed using the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) hardware testbed onboard the International Space Station (ISS). First, this chapter introduces the SPHERES tested. Then, it describes the implementation of the reconfiguration maneuver experiments on SPHERES onboard ISS. Finally, the experimental data are shown and compared to the theoretical results.

5.1 SPHERES Background

The SPHERES testbed was developed by the MIT Space Systems Laboratory (SSL) to primarily perform true laboratory experiments onboard the ISS in a micro-gravity environment. It can also operate onboard NASA’s Reduced Gravity Aircraft (both the C-9B and the former KC-135), and on a flat floor (similar to the one in NASA Marshall Space Flight Center (MSFC) Flight Robotics Laboratory) [61].

The SPHERES testbed provides a cost-effective, long duration, and easily reconfigurable environment that allows the development, validation, and maturation of spacecraft formation flying and autonomous rendezvous and docking [41]. The testbed is designed to give the opportunity to multiple scientists to validate new
Theories, rather than just meeting the traditional quantitative requirements for a specific space mission [62]. A SPHERES satellite is shown in Figure 5-1. SPHERES key characteristic lies in its modularity in the algorithm development. Each software module contains an algorithm to achieve a simple task. The guidance, navigation and control (GN&C) architecture used on the SPHERES testbed includes the following modules: 1) state estimation, 2) control, 3) mission and vehicle management, and 4) fault detection, isolation and recovery. Ref. [63] provides a detailed description of each of these modules. Another important feature of SPHERES is the flexibility of the interface with the hardware. This feature facilitates the testing of algorithms that originate from different research areas ranging from formation flight to autonomous docking [61]. Prior to sending SPHERES algorithms to the ISS, the scientists conduct tests on the SPHERES flat table testbed in the MIT SSL laboratory. The flat table testbed uses the same flight hardware that is onboard ISS, but the SPHERES are mounted on air carriages to float on it [41].

Each SPHERES satellite has a diameter of 0.25 m, a mass of 4.2 kg, and can produce a 0.22 N thrust in each axial direction through twelve thrusters that provide
controllability in all six degrees of freedom, thus enabling both torque and translation control [61].

The SPHERES testbed is currently in orbit onboard ISS in the US laboratory. Currently there are three SPHERES microsatellites onboard the ISS. The first experiments were performed on May 20, 2006, and they successfully tested simple attitude slew maneuvers, a docking maneuver with a fixed target, and a position-hold experiment [63]. Then followed several other SPHERES test sessions (eight in total) that demonstrated a series of complex maneuvers over the last two years.

5.2 Reconfiguration Maneuvers on the SPHERES Testbed

The two-stage path planning algorithm introduced in this thesis was demonstrated on the SPHERES testbed onboard the ISS. At the time of these experiments, only two SPHERES were in orbit onboard the ISS, thus the maneuvers tested consisted of one and two satellite experiments. The SPHERES testbed did not contain a linear or nonlinear programming solver, so instead of solving the path planning problem online, the solution for the reconfiguration maneuver problem was computed off-line, and encoded onboard the SPHERES microsatellites as a series of waypoints. Each waypoint consisted of a state vector as defined in (2.2). A PD controller combined with pulse-width modulators were used by the SPHERES to closely track the waypoints, which were expressed in a coordinate frame attached to the ISS. Note that the maneuvers were first tested on the SPHERES flat table to verify and validate the flight code prior to sending it to the ISS.

The dynamics used in the design of the ISS reconfiguration maneuvers are defined in (2.5) and (2.7). Notice that the translation dynamics are expressed as double integrators, since the effects of the Earth gravity gradient are minimal due the sufficiently small operating area inside the ISS. The design of the maneuvers takes into consideration the danger of losing metrology near the edges of the working area. Therefore
all maneuvers are planned inside a "virtual box" centered at the middle of the ISS SPHERES working area and side lengths 1.5 m. Note that some "virtual" fixed obstacles are considered in the design of the maneuvers for the purpose of making the reconfiguration problem more challenging.

The astronauts use a NASA laptop computer onboard the ISS as a ground station to transmit commands to the satellites and record telemetry [61]. At the beginning of each experiment, the astronauts follow the initialization instructions that specify the initial positions and attitudes of the SPHERES. After waiting for the estimator to converge, the SPHERES satellites move to their initial configuration or waypoint. They then closely follow the trajectories until they reach their final configurations. States, state errors, and thruster data are recorded and sent to the NASA laptop on the fly.

At the time of the experiments, the RRT-GPM technique introduced in this thesis was not implemented mainly because the GPOCS software package [43] was not released. But the RRT-LPM approach described in Section 4.2.1 was already coded since the DIDO software [57] was available. In addition, the RRT-LS approach described in Section 4.1.1 and based on the work done in [10] was also implemented. So the RRT-LPM and RRT-LS approaches were used to design the SPHERES ISS reconfiguration maneuvers. The experiments designed in Sections 4.1.2, 4.1.3, 4.2.2, and 4.2.3 were performed onboard ISS. The next section illustrates the flight results returned by these experiments.

### 5.3 Flight Experiment Results

The ISS experiments are divided into two sections: RRT-LS and RRT-LPM. The first section shows the results of the ISS experiments that are based on the RRT-LS approach (Section 4.1.1). These experiments were performed on March 17, 2007 during the 6th SPHERES ISS test session. Figure 5-2 shows a picture of two SPHERES onboard ISS during the 6th test session executing a reconfiguration maneuver. The second section displays the results of the ISS experiments that were designed using
the RRT-LPM approach (Section 4.2.1), and were performed on April 27, 2007 during the 8th SPHERES ISS test session. Figures 5-3 and 5-4 are snapshots from the videos of the two experiments performed onboard the ISS using the RRT-LPM approach.

5.3.1 RRT-LS Experiment Results

The examples described in Sections 4.1.2 and 4.1.3 were performed onboard ISS during the 6th SPHERES ISS test session. But in the design of the ISS experiments, the original cost defined in (4.6) was used instead of the linearization of the quadratic cost discussed in Section 4.1.2. So theoretical results will be shown along with data from the ISS experiments.

Section 4.1.2 is a single satellite maneuver that includes a fixed obstacle and two stay outside constraints. First the theoretical results are shown in Figures 5-5 to 5-7. The theoretical cost of the maneuver is 0.363. Then the results of the ISS experiment are shown in Figures 5-8 to 5-13. Figure 5-8 is the trajectory followed by
Figure 5-3: Snapshots from the video of a SPHERES performing a reconfiguration maneuver onboard the ISS using the RRT-LPM approach (Courtesy of NASA).

Figure 5-4: Snapshots from the video of two SPHERES performing a reconfiguration maneuver onboard the ISS using the RRT-LPM approach (Courtesy of NASA).
the SPHERES during the reconfiguration maneuver. Figures 5-9 and 5-11 show the states of the spacecraft and the input controls over time. Figure 5-10 displays the error between the desired states and the actual states of the SPHERES microsatellite. The trajectory followed in this experiment satisfies the obstacle avoidance and pointing restriction constraints as shown in Figures 5-12 and 5-13. The ISS cost of this maneuver is 0.456.

Section 4.1.3 is a two-satellite satellite maneuver that includes a fixed obstacle and an inter-spacecraft pointing constraint. The theoretical results are first shown in Figures 5-14 to 5-16. The theoretical cost of the maneuver is 1.012. Then, the ISS results of this experiment are shown in Figures 5-17 to 5-22. The trajectories followed by the two SPHERES during this experiment onboard ISS is shown in Figure 5-17. Figures 5-18 and 5-19 show the states of the spacecraft and the input controls over time. Figure 5-20 displays the error between the desired states and the actual states for the two microsatellites. Figure 5-21 shows that the inter-spacecraft pointing constraint is satisfied during the entire duration of the maneuver. The obstacle avoidance constraint, shown in Figure 5-22, is violated by spacecraft 1 between \(t = 21\) s and \(t = 23\) s by less than 1 cm. But it is otherwise satisfied by both spacecraft. Finally, the cost of this maneuver computed out of the ISS data is 1.353.
Figure 5-5: Theoretical final trajectory of the single spacecraft maneuver designed using the RRT-LS approach.

Figure 5-6: Theoretical states of the spacecraft over time for the single spacecraft maneuver designed using the RRT-LS approach.
Figure 5-7: Theoretical input controls over time for the single spacecraft simulated maneuver designed using the RRT-LS approach.
Figure 5-8: Trajectory of the spacecraft during the ISS single spacecraft maneuver designed using the RRT-LS approach.

Figure 5-9: States of the spacecraft over time for the ISS single spacecraft maneuver designed using the RRT-LS approach.
Figure 5-10: State errors of the spacecraft over time for the ISS single spacecraft maneuver designed using the RRT-LS approach.

Figure 5-11: Input controls over time for ISS single spacecraft maneuver designed using the RRT-LS approach.
Figure 5-12: Cosines of the angles between the ranging device and the restriction pointing vectors during the ISS single spacecraft maneuver designed using the RRT-LS approach.

Figure 5-13: Distance between the center of the spacecraft and the center of the obstacle during the ISS single spacecraft maneuver designed using the RRT-LS approach.
Figure 5-14: Theoretical trajectories of the two-spacecraft maneuver designed using the RRT-LS approach.
Figure 5-15: Theoretical states of the spacecraft over time for the two-spacecraft maneuver designed using the RRT-LS approach.
Figure 5-16: Theoretical input controls for the spacecraft over time during the two-spacecraft maneuver designed using the RRT-LS approach.
Figure 5-17: Trajectories of the spacecraft during the ISS two-spacecraft maneuver designed using the RRT-LS approach.
Figure 5-18: States of the spacecraft over time for the two-spacecraft ISS maneuver designed using the RRT-LS approach.
Figure 5-19: Input controls for the spacecraft over time during the two-spacecraft ISS maneuver designed using the RRT-LS approach.
Figure 5-20: State errors over time for the two-spacecraft spacecraft ISS maneuver designed using the RRT-LS approach.
Figure 5-21: Cosines of angles between ranging device vector and relative position of both spacecraft during the two-spacecraft ISS maneuver.

Figure 5-22: Distance between the centers of the spacecraft and the center of the obstacle during the two-spacecraft ISS maneuver designed using the RRT-LS approach.
5.3.2 RRT-LPM Experiment Results

The examples described in Sections 4.2.2 and 4.2.3 were performed onboard ISS during the 8th SPHERES ISS test session.

Section 4.2.2 is a single satellite maneuver that includes a fixed obstacle and two "stay outside" constraints. The results of this experiment are illustrated in Figures 5-23 to 5-28. Figure 5-23 is the trajectory followed by the SPHERES during the reconfiguration maneuver. Figures 5-24 and 5-26 show the states of the spacecraft and the input controls over time. Figure 5-25 displays the error between the desired states and the actual states of the SPHERES microsatellite. It is seen that the trajectory followed in this experiment satisfies the obstacle avoidance and pointing restriction constraints as shown in Figures 5-28 and 5-27. Finally, the cost of this maneuver is 0.00815.

The next experiment designed in Section 4.1.3 is a two-satellite satellite maneuver that includes a fixed obstacle and an inter-spacecraft pointing constraint. The results of this experiment are shown in Figures 5-29 to 5-34. The trajectories followed by the two SPHERES during this experiment onboard ISS is shown in Figure 5-29. Figures 5-30 and 5-31 show the states of the spacecraft and the input controls over time. Figure 5-32 displays the error between the desired states and the actual states for the two microsatellites. Figure 5-33 shows that the obstacle avoidance constraint is satisfied during the entire maneuver. The inter-spacecraft pointing constraint is violated between between $t = 16$ s and $t = 24$ s, but it is satisfied everywhere else outside this window of time. Finally, this maneuver has a cost of 0.0343.
Figure 5-23: Trajectory of the spacecraft during the ISS single spacecraft maneuver designed using the RRT-LPM approach.

Figure 5-24: States of the spacecraft over time for the ISS single spacecraft maneuver designed using the RRT-LPM approach.
Figure 5-25: State errors of the spacecraft over time for the ISS single spacecraft maneuver designed using the RRT-LPM approach.

Figure 5-26: Input controls over time for ISS single spacecraft maneuver designed using the RRT-LPM approach.
Figure 5-27: Cosines of the angles between the ranging device and the restriction pointing vectors during the ISS single spacecraft maneuver designed using the RRT-LPM approach.

Figure 5-28: Distance between the center of the spacecraft and the center of the obstacle during the ISS single spacecraft maneuver designed using the RRT-LPM approach.
Figure 5-29: Trajectories of the spacecraft during the ISS two-spacecraft maneuver designed using the RRT-LPM approach.
Figure 5-30: States of the spacecraft over time for the two-spacecraft ISS maneuver designed using the RRT-LPM approach.
Figure 5-31: Input controls for the spacecraft over time during the two-spacecraft ISS maneuver designed using the RRT-LPM approach.
Figure 5-32: State errors over time for the two-spacecraft spacecraft ISS maneuver designed using the RRT-LPM approach.

(a) Spacecraft 1

(b) Spacecraft 2
Figure 5-33: Cosines of angles between ranging device vector and relative position of both spacecraft during the two-spacecraft ISS maneuver.

Figure 5-34: Distance between the centers of the spacecraft and the center of the obstacle during the two-spacecraft ISS maneuver designed using the RRT-LPM approach.
5.3.3 Discussion

Below are some observations and discussion related to the results of the ISS experiments.

- The SPHERES microsatellites succeeded in following the planned trajectories achieving the desired reconfiguration maneuvers. The obstacle and pointing constraints were satisfied in the majority of the tests. The violations that occurred were limited to a short period of time compared to the duration of the maneuver. They are believed to be caused by measurement errors. Another reason could be related to the limitation of PD controllers which are known to be incapable of completely removing steady state errors [64].

- The state error graphs show that the maximum error between the theoretical and actual ISS values is less than 12 cm in position. Furthermore, the maximum error in the attitude variables in, in general, less than 12 degrees. However, Figure 5-25 presents a spike in the attitude error that reaches 17 degrees. A possible explanation is that this spike is due to a measurement error due to sensor noise. It can also be caused by multipath in the ultrasound signal transmission. For this same reason, a PD controller was used to perform the reconfiguration maneuver tests on ISS, instead of a PID controller. If a PID controller was used as the onboard controller, “long” spikes in the measurement errors could have caused the integrator in the PID to windup [64]. Recently, a filter innovation threshold has been added to the SPHERES estimator [61]. A measurement with an innovation above a specified threshold is rejected because it is a sign of non-coherence. Thus future reconfiguration maneuver experiments will use a PID controller, and the state errors are expected to be reduced significantly.

- The costs of the ISS maneuvers are summarized in Table 5.1. They are shown next to the theoretical results computed in Chapter 4. The costs of the ISS experiments that used the RRT-LS approach are relatively close to the theoretical values: the cost is 26% higher for the single satellite experiment, and 33%
Table 5.1: Comparison of total costs of the ISS reconfiguration maneuvers against theoretical results.

<table>
<thead>
<tr>
<th>Maneuver</th>
<th>Theoretical Results</th>
<th>ISS Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RRT-LS</td>
<td>RRT-LPM</td>
</tr>
<tr>
<td>1 s/c</td>
<td>0.363</td>
<td>0.00213</td>
</tr>
<tr>
<td>2 s/c</td>
<td>1.012</td>
<td>0.00986</td>
</tr>
</tbody>
</table>

The difference is mainly due the extra fuel that the SPHERES has to consume in order to correct the errors in the states while following the nominal path.

Comparatively, the ISS experiments designed using the RRT-LPM approach have costs considerably higher than the theoretical ones: the cost is 3.8 times higher in the single satellite experiment, and 3.5 times higher for the two-satellite experiment. The reason for this difference is directly related to the way the thrusters work on SPHERES. In fact, they are ON/OFF type and cannot produce variable thrust levels. Therefore the force and torque commands given by the controller are converted to thruster ON/OFF using a pulse-width modulator \[1\].

Figure 5-35 shows the mechanisms of this conversion. The thrust impulse produced is centered in the control period. It is designed not to occupy more than 25% of each control period, since the other 75% of the time is reserved for taking measurements. Let us assume that the commanded thrust has an amplitude \(A\), and that the control period is called \(T_c\). Then the cost of a control period, defined as the square of the trust multiplied by the time period (for the RRT-LPM case), is equal to \(A^2 \times T_c\) for the commanded thrust, but is equal to \((4 \times A)^2 \times T_c/4 = 4 \times A^2 \times T_c\) for the produced thrust. Therefore, the produced cost is 4 times the commanded cost. This explains the differences between the theoretical results and ISS results for the maneuvers using the RRT-LPM approach. This behavior is apparent on the RRT-LPM experiments because of the continuous type of controls that the RRT-LPM approach produces. But notice that the ISS costs of the RRT-LS experiments are less affected by this
thrust conversion because of the “impulse-like” type of the controls that the RRT-LS approach returns (See Section 4.1.1). In fact, the impulse-like controls are similar in nature to the discretized thrusts produced by the SPHERES. Another important reason that explains the smaller error in the RRT-LS costs is that the cost function used to design the RRT-LS maneuvers is the total fuel consumed, compared to the total energy in the RRT-GPM case. The total fuel is less affected by the thrust conversion mechanism since it’s the sum of the absolute value of the controls.

The ISS experiments achieved the main goal behind them: the application of the two-stage reconfiguration maneuvers design to real satellites in a microgravity environment, thus validating the algorithms used to design them. Another goal that was also met was to learn from these experiments in order to improve the two-stage algorithms, and apply the improvements on future ISS reconfiguration maneuvers.

The main lessons learnt from the ISS experiments and recommendation for future improvements are listed below:

- The thruster levels commanded by the controller do not correspond to the levels computed in the design process. This observation is due to the thruster conversion shown in Figure 5-35. The knowledge of this conversion should be added to the two-stage reconfiguration design to improve the results. It might be also beneficial to try designing the RRT-LPM maneuvers using the total fuel.
consumption as the cost to optimize rather than the total energy, and compare
the results.

- The measurement errors of the SPHERES flying on ISS are not negligible. An
innovation filter analysis developed in Ref. [61] helps in reducing these errors. Other fault detection, identification and recovery (FDIR) techniques might be
needed to reduce these errors further.

- The PD controller causes state errors. Replacing the PD controller by a PID
controller will be essential in reducing the levels of error observed. More “aggress-
vie” maneuvers could be then performed, including closer collision avoidance
experiments.

- The SSL flat table is an invaluable facility to repetitively test and validate the
algorithms in a low cost environment before sending them to the ISS. It removes
the risk of facing implementation errors onboard the ISS. It was the reason why
the reconfiguration maneuvers performed onboard the ISS did not face any ma-
jor hardware or software error. Since future ISS SPHERES reconfigurations will
be more complex (they will include up to three SPHERES), the flat table facil-
ity will become even more important to ensure the success of the experiments
onboard the ISS.

- Maneuvers designed using the RRT-GPM approach and including up to three
SPHERES should be performed in the coming ISS test sessions. In the longer
term, the planning of the reconfiguration maneuvers should be done onboard
SPHERES in real-time. To enable such a great capability, a linear and nonlinear
solvers should be implemented on SPHERES, along with more memory and
preferably a faster microprocessor.

5.4 Summary

This chapter presented the results of the reconfiguration maneuvers that were per-
formed onboard the International Space Station using the SPHERES testbed. Flight
data were plotted and compared to the theoretical results. The success of these experiments validate the results of the two-stage algorithms, the RRT-LS and the RRT-LPM techniques, which are introduced in Chapter 4. Several lessons were learnt from these tests, and will be essential in improving future ISS reconfiguration experiments.
Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis has introduced a two-stage path planning approach that solves spacecraft reconfiguration maneuver problems. These problems are challenging because they include nonlinear attitude dynamics, difficult non-convex constraints, and coupling of the multiple spacecraft states in the constraints. This two-stage technique extends the original ideas in Ref. [10] to improve the second step by using a Gauss pseudospectral method (GPM) [39], a technique that has become recently popular among direct numerical methods in solving optimal control problems.

Chapter 2 describes the two-stage technique. The first stage consists of solving a simplified path planning problem without differential constraints using bidirectional Rapidly-exploring Random Trees [35]. The output of the first stage is then augmented in a ”transition” step with feasible dynamics. The result is a feasible initial solution that is given to a second stage to be improved. The second stage is formulated as an optimal control problem based on a Gauss pseudospectral method, and solved using the GPOCS software package [43] along with the SNOPT nonlinear solver [48].

Chapter 3 presents the improvement that initialization techniques based on RRTs bring to reconfiguration maneuver problems solved using pseudospectral methods. It shows that such a technique is able to compute initial feasible guesses that 1) considerably reduces the computation times of the solutions, and 2) makes it possible
to solve more complex reconfiguration maneuvers. The reason is that a feasible initial guess helps the nonlinear solver used in pseudospectral methods to converge to an optimal solution faster and more reliably. This improvement is illustrated with a series of examples including up to five spacecraft, that are solved with and without an initial RRT guess. This initialization technique can also be of benefit for other applications that rely on pseudospectral methods in their solution approach.

In Chapter 4, the RRT-GPM approach is compared to two other two-stage trajectory planning approaches: RRT-LS and RRT-LPM. All three approaches use the same RRT planner in their first stage. But in their second stage, RRT-LS uses a linear smoother (LS) while RRT-LPM relies on a Legendre pseudospectral method (LPM). The results show that the RRT-GPM approach has worse computation times than RRT-LS, but its approach is considerably more accurate and achieves far better final costs. RRT-GPM has been also shown to outperform the RRT-LPM approach both in the computation times and in the optimality of the solution.

Finally, Chapter 5 presents the results of reconfiguration maneuvers performed onboard the International Space Station (ISS) using the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) hardware testbed. The maneuvers are examples of reconfiguration maneuvers designed in Chapter 4 using the RRT-LS and RRT-LPM approach. The RRT-GPM was not used in the design of the maneuvers because the GPOCS software was not available at the time of the experiments. The experiments consist of one and two spacecraft maneuvers including fixed obstacles and pointing constraints. The flight data are analyzed and compared to the designed trajectories. The results show that the SPHERES succeeded in general in performing the planned reconfiguration maneuvers. The sources of violations that happen for a short period of time are also explained. The success of these experiments validate the results of the two-stage algorithms, RRT-LS and RRT-LPM, which are presented in Chapter 4. This chapter also discusses the lessons learnt from the ISS SPHERES experiments and the recommendations for future test sessions.
6.2 Recommendations for Future Work

Future work divides in to two different areas: the RRT-GPM solution approach and the real-time path planning of spacecraft reconfiguration maneuvers.

6.2.1 Future Work for the RRT-GPM Approach

Advances in this area should begin with the improvement of RRT-GPM approach to account for moving obstacles. The RRT planner would have to take into account the notion of time in its random sampling process [35]. This addition would increase the computation time, but it would enable a solution technique for more dynamic problems. Another possible continuation is the extension of the RRT-GPM problem formulation to cover multiple-phase reconfiguration maneuvers. The RRT planner would be adjusted to solve a series of planning phases where the boundaries between each phase can be fixed or moving in space and/or time. The GPM phase could take advantage of the GPOCS software package that solves multiple-phase optimal control problems. Currently, the GPOCS relies on the SNOPT nonlinear solver in its solution approach. SNOPT is based on a sequential quadratic programming (SQP) method. Therefore another continuation is solving the GPM phase with a solver that is based on a different nonlinear method approaches, like the interior point method used in the LOQO solver [65].

6.2.2 Future Work for Real-Time Path Planning of Reconfiguration Maneuvers

Real-time path planning is one of the hottest topics in the field of Aerospace Engineering right now. Chapter 5 has presented results of reconfiguration maneuvers that were computed offline, and flown in the ISS onboard the SPHERES microsatellites as a series of waypoints the ISS. Future work should implement the two-stage path planning algorithms discussed in this thesis onboard the SPHERES, so the path planning is performed in real-time. RRT-GPM requires a nonlinear solver in its solu-
tion process, therefore in order to implement it online, the SPHERES testbed has to be upgraded with a nonlinear solver capability. SPHERES would also require more memory and processing power to be able to solve nonlinear programs in a reasonable amount of time suitable for online path planning. A first step in the online implementation of RRT-GPM would be trying to implement the RRT-LS algorithm, which only requires a linear solver, a capability that can be more easily added to the SPHERES hardware. The goal should be to achieve a robust path planner that plans in real-time reconfiguration maneuvers in a consistent and reliable fashion.
Bibliography


International Symposium on Artificial Intelligence, Robotics, and Automation for Space (iSAIRAS), July 1997.


