Two-Stage Path Planning Approach for Designing Multiple Spacecraft Reconfiguration Maneuvers

Georges S. Aoude, Jonathan P. How and Ian M. Garcia

Abstract— **The paper 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 paper also places emphasis on the importance of the initialization step in pseudospectral methods in order to decrease their computation times and enable the solution of a more complex class of problems. Several examples are presented and discussed.**

I. INTRODUCTION

The Terrestrial Planet finder (TPF) [1], the Laser Interferometer Space Antenna Project (LISA) [2], the Micro-Arcsecond X-ray Imaging Mission (MAXIM) [3], the System F6 Program to demonstrate a fractionated spacecraft approach [4], 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 [5]. For example, the proposed TPF observatory consists of multiple spacecraft carrying infrared telescopes[6], [1]. 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 planning algorithms, autonomous fault detection and recovery, and on the high level mission management [7].

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 [8].

This paper 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). 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.) [8]. 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 force coupling between the position and attitude the spacecraft, making it impossible

G. S. Aoude, Ph. D. Candidate, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, gaoude@mit.edu

J. P. How, Professor, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, jhow@mit.edu

I. G. Garcia, Technical Staff, Space Navigation Group, MIT, Cambridge, MA 02139, USA, igarcia@draper.com

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.

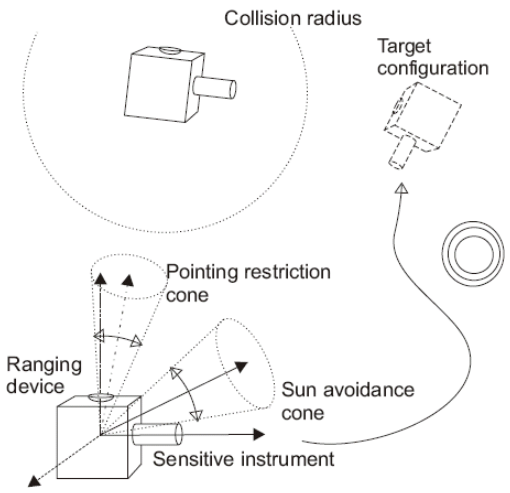


Fig. 1. The formation reconfiguration problem [9]

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 [10], 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.

II. 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. [11], [12] 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 [13], [14]. 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 [15]. 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 [16], [17], and their incremental counterparts the Rapidly-exploring Random Trees (RRT) algorithms. But the application of RRTs on spacecraft reconfiguration problems was limited to 1) a problem involving a single spacecraft with no pointing constraints [18], 2) a problem considering the attitude maneuver of one spacecraft [15], and 3) a multi-spacecraft reconfiguration problem that solves for the translation trajectory only [19]. 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 [20], [21]. This approach consists of a two-stage planning algorithm, similar to the one developed in this research. 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 [22], [23], [24]. 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 [25]. Ref. [23] 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. [26], 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 process is to be efficient, the algorithm must be chosen with care. The technique presented in this paper improves the performance of pseudospectral method based problems 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 bidirectional RRTs [21].

III. Solution Concepts

This paper presents a two-stage path planning algorithm to solve the problem described in Section IV. The first stage, discussed in Section III-B, is based on Rapidly-exploring Random Trees (RRTs), a randomized planning technique that has been very popular recently. Then, Section III-C 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. [9] to improve the second step

by using a specific pseudospectral method, called the Gauss pseudospectral method (GPM).

A. 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 [27], which has been traditionally implemented using two common approaches: "two-stage" planning and "state-space" formulation [28], [27]. Unlike the "state-space" approach, where the dynamic constraints are taken into account from the start of the algorithm [29], [30], 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 [31], [20]. This paper develops a two-stage approach for solving reconfiguration maneuvers of multiple spacecraft. Examples of complex maneuvers including up to five spacecraft illustrate this approach.

B. Rapidly Exploring Random Trees

The first stage of the two-stage algorithm developed in this paper 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) [16]. Rapidly exploring Random Trees (RRTs), a recent variant of PRMs introduced in Refs. [32], [33], 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 [32]. We emphasize two of them: 1) their expansion is heavily biased towards unexplored areas of the configuration space (*e.g.,* see Figure 2) 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 [34]. This paper 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 [21]. Section V-A describes this method in more detail.

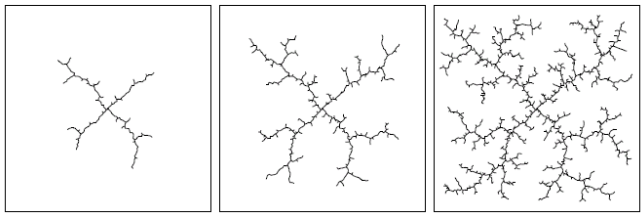


Fig. 2. Example of RRT expansion starting from center of square [32]

C. Pseudospectral Methods

The second stage of the planning algorithm developed in this paper 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 [35].

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 [36], [37], 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 [38]. 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 paper 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 [22], [39]. Section V-C describes the Gauss pseudospectral method in its most current form.

IV. Problem Formulation

The general reconfiguration problem resides in finding a trajectory of N spacecraft from time 0 to time T . Let $\boldsymbol{p}_i(t)$

be a point of the trajectory of a single spacecraft at time t. This point consists of

$$\boldsymbol{p}_i(t) = [\boldsymbol{x}_i(t), \boldsymbol{u}_i(t)], \tag{1}$$

where $\boldsymbol{x}_i$(t) and $\boldsymbol{u}_i$(t) represent the state and control inputs at each time t, respectively,

$$\boldsymbol{x}_i(t) = [\boldsymbol{r}_i(t), \dot{\boldsymbol{r}}_i(t), \boldsymbol{w}_i(t), \boldsymbol{\sigma}_i(t)], \tag{2}$$

$$\boldsymbol{u}_i(t) = [\boldsymbol{f}_i(t), \boldsymbol{\tau}_i(t)], \tag{3}$$

and where i $\in 1 \dots N$ indicates the spacecraft. $\boldsymbol{r}_i$(t) $\in \mathbb{R}^3$ is the position of its center, $\dot{\boldsymbol{r}}_i$(t) $\in \mathbb{R}^3$ is its velocity, $\boldsymbol{w}_i$(t) $\in \mathbb{R}^3$ its angular velocity, and $\boldsymbol{\sigma}_i$(t) $\in \mathbb{R}^3$ is its attitude representation in modified Rodrigues parameters (MRP) [40]. All these variables are measured with respect to a local inertially fixed frame. $\boldsymbol{f}_i$(t) $\in \mathbb{R}^3$ represents the control input force, and $\boldsymbol{\tau}_i$(t) $\in \mathbb{R}^3$ the control input torque. Therefore,

$$\boldsymbol{p}(t) = [\dots, \boldsymbol{p}_i(t), \dots], \tag{4}$$

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

$$\begin{bmatrix} \dot{\boldsymbol{r}}_i(t) \\ \ddot{\boldsymbol{r}}_i(t) \end{bmatrix} = \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_i(t) \\ \dot{\boldsymbol{r}}_i(t) \end{bmatrix} + \begin{bmatrix} 0_{3\times3} \\ \frac{I_{3\times3}}{M} \end{bmatrix} \boldsymbol{f}_i(t) \tag{5}$$

where M $\in \mathbb{R}$ is the mass, assumed to be the same for all spacecraft for simplicity. $I_{3\times3}$ is the 3×3 identity matrix, and $0_{3\times3}$ is the 3×3 zero matrix.

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

$$\dot{\boldsymbol{\sigma}}_i(t) = R(\boldsymbol{\sigma}_i(t))\boldsymbol{w}_i(t) \tag{6}$$

$$\begin{aligned} J\dot{\boldsymbol{w}}_i(t) &= -\boldsymbol{w}_i(t) \times J\boldsymbol{w}_i(t) + \boldsymbol{\tau}_i(t) \\ &= -S(\boldsymbol{w}_i(t))J\boldsymbol{w}_i(t) + \boldsymbol{\tau}_i(t) \end{aligned} \tag{7}$$

where $J \in \mathbb{R}^3$ is the spacecraft constant inertia matrix, considered to the same for all spacecraft for simplicity. S $\in \mathbb{R}^{3\times3}$ is the skew-symmetric matrix representing the cross product operation

$$S(\boldsymbol{a}) \triangleq [\boldsymbol{a}\times] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}, \quad \forall \boldsymbol{a} \in \mathbb{R}^3 \tag{8}$$

The Jacobian matrix $R \in \mathbb{R}^{3\times3}$ for MRP attitude representation is given by [40]

$$R(\boldsymbol{\sigma}_i) = \frac{1}{4}\left[(1 - \boldsymbol{\sigma}_i^T\boldsymbol{\sigma}_i)I_{3\times3} + 2S(\boldsymbol{\sigma}_i) + 2\boldsymbol{\sigma}_i\boldsymbol{\sigma}_i^T\right] \tag{9}$$

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

$$\|\boldsymbol{r}_i(t) - \boldsymbol{r}_j(t)\| \geq \boldsymbol{R}_{ij} \tag{10}$$

for $i, j \in 1 \dots N$, $i \neq j$, and $\boldsymbol{R}_{ij}$ 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

$$\|\boldsymbol{r}_i(t) - \boldsymbol{l}_o(t)\| \geq \boldsymbol{R}_{io} \tag{11}$$

for every obstacle o, and for $i \in 1 \dots N$. $\boldsymbol{l}_o$ is the position of the center of obstacle o, and $\boldsymbol{R}_{io}$ 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

$$\boldsymbol{z}_k^T\boldsymbol{y}_k(t) \leq \cos\theta_k \tag{12}$$

for every stay outside pointing constraint k. This constraint ensures that the spacecraft vector $\boldsymbol{y}_k$ remains at an angle greater than $\theta_k \in [0, \pi]$ from the inertial vector $\boldsymbol{z}_k$. The vector $\boldsymbol{y}_k$ represents the body vector $\boldsymbol{y}_{kB}$ in the inertial coordinate frame. The transformation of coordinates is given by

$$\boldsymbol{y}_k(t) = Rot^{-1}(\boldsymbol{\sigma(t)})\boldsymbol{y}_{kB} \tag{13}$$

where $Rot(\boldsymbol{\sigma}(t))$ is the rotation matrix representation of the MRP attitude vector $\boldsymbol{\sigma}(t)$, which can be written as [40]

$$Rot(\boldsymbol{\sigma}) = I + \frac{4(1 - \boldsymbol{\sigma}^T\boldsymbol{\sigma})}{(1 + \boldsymbol{\sigma}^T\boldsymbol{\sigma})^2}S(\boldsymbol{\sigma}) + \frac{8}{(1 + \boldsymbol{\sigma}^T\boldsymbol{\sigma})^2}S(\boldsymbol{\sigma})^2 \tag{14}$$

where S is the matrix defined in (8). It is assumed that $\boldsymbol{y}_{KB}$ and $\boldsymbol{z}_k$ are fixed vectors *i.e.,* independent of time t.

The absolute stay inside constraints only change the sign of the inequality of (12). They can be written as

$$\boldsymbol{z}_k^T\boldsymbol{y}_k(t) \geq \cos\theta_k \tag{15}$$

The inter-spacecraft relative stay outside constraints are given by

$$\hat{\boldsymbol{r}}_{ij}^T(t)\boldsymbol{y}_k(t) \leq \cos\theta_k \tag{16}$$

where $\boldsymbol{y}_k$(t) and θ_k are the same as defined above, and

$$\hat{\boldsymbol{r}}_{ij}(t) = \frac{\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)}{\|\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)\|} \tag{17}$$

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

$$\hat{\boldsymbol{r}}_{ij}^T(t)\boldsymbol{y}_k(t) \geq \cos\theta_k \tag{18}$$

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

$$\boldsymbol{x}_i(0) = \boldsymbol{x}_{is} \tag{19}$$

$$\boldsymbol{x}_i(T) = \boldsymbol{x}_{if} \tag{20}$$

where $\boldsymbol{x}_{is}$ represents the state corresponding to the specified starting condition, and $\boldsymbol{x}_{if}$ to the specified final condition $\forall i \in 1 \dots N$.

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

$$\boldsymbol{x}_{min} \leq \boldsymbol{x}_i(t) \leq \boldsymbol{x}_{max} \tag{21}$$

$$\boldsymbol{u}_{min} \leq \boldsymbol{u}_i(t) \leq \boldsymbol{u}_{max} \tag{22}$$

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 [41].

The objective is to minimize the total energy of the formation

$$\boldsymbol{J} = \sum_{i=1}^{N} \int_0^T \|\boldsymbol{f}_i(t)\|^2 + \|\boldsymbol{\tau}_i(t)\|^2 dt \tag{23}$$

Minimizing the total energy consumption of a formation of spacecraft is an objective for many space missions [42], [43]. Furthermore, the energy is in general directly related to the fuel consumption. So minimizing energy energy leads to less fuel consumption.

V. Solution Approach

A. 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. [21]. It is reproduced here for clarity. The original RRT algorithm was developed by Lavalle [34].

Algorithm 1 RRT-BIDECTIONAL ($\boldsymbol{p}_i$,$\boldsymbol{p}_f$)

```
T_a.init(p_i); T_b.init(p_f);
for j ← 1 to K do
    p_n ← NEAREST(T_a, α(j))
    p_s ← POTENTIAL-CONNECT(p_n, α(j))
    if p_s ≠ p_n then
        T_a.ADD-VERTEX(p_s)
        T_a.ADD-EDGE(p_n, p_s)
        p̂_n ← NEAREST(T_b, p_s)
        p̂_s ← POTENTIAL-CONNECT(p̂_n, p_s)
        if p̂_s ≠ p̂_n then
            T_b.ADD-VERTEX(p̂_s)
            T_b.ADD-EDGE(p̂_n, p̂_s)
        end if
        if p̂_s = p_s then
            return Solution
        end if
    end if
    SWAP(T_a, T_b)
end for
return Failure
```

In Algorithm 1, $\boldsymbol{T}_a$ and $\boldsymbol{T}_p$ represent trees having a composite trajectory point $\boldsymbol{p}$ at each node (4). $\boldsymbol{T}_a$ starts from the initial point and $\boldsymbol{T}_b$ starts from the final point of the goal trajectory. At each node, the points $\boldsymbol{p}$ 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 $\boldsymbol{T}_a$ with the minimum distance to the point $\alpha(i)$ is found by calling NEAREST($\boldsymbol{T}_a$, $\alpha(i)$). Distance in this context represents a weighted summation of rotation and translation. It can be written as

$$d(\boldsymbol{p}_1, \boldsymbol{p}_2) = \sum_{i=1}^{N} \|\boldsymbol{r}_{1,i} - \boldsymbol{r}_{2,i}\| + K_a \angle(\boldsymbol{q}_{1,i}, \boldsymbol{q}_{2,i}) \tag{24}$$

where $\angle(\boldsymbol{q}_{1,i}, \boldsymbol{q}_{2,i})$ represents the angle of an eigen-axis rotation between attitude $\boldsymbol{q}_{1,i}$ and $\boldsymbol{q}_{2,i}$ 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 $d(\boldsymbol{p}_1, \boldsymbol{p}_2)$ where the obstacle avoidance, restricted pointing, and other constraints are represented by inequality and equality constraints [21]. 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. 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.

Algorithm 2 POTENTIAL-CONNECT($\boldsymbol{p}$, $\boldsymbol{p}_f$)

```
for j ← 1 to K do
    Solve nonlinear program:
      min_dp d(p + dp, p_f)
      subject to
      g_min,i ≤ g_i(p + dp) ≤ g_max,i, ∀i
      ‖dp‖ ≤ ε
    ▶ End of nonlinear program
    p ← p + dp
end for
return p
```

POTENTIAL-CONNECT tries to connect $\boldsymbol{p}$ to $\boldsymbol{p}_f$ by moving in small dp increments. These dp increments are restricted to be smaller in norm than ϵ 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.

So the solution of the first stage consists of a sequence of points from the initial point $\boldsymbol{p}_i$ to the final point $\boldsymbol{p}_f$. 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 1 and Algorithm 2, the interested reader is encouraged to consult references [21] and [9].

B. 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 $\boldsymbol{p}^{inter}$ half way between every pair of nodes. First assume that the problem consists of only one spacecraft. To propagate the dynamics to $\boldsymbol{p}^{inter}$, the spacecraft is assumed to accelerate under a constant input force $\hat{\boldsymbol{f}}$ and a constant input torque $\hat{\boldsymbol{\tau}}$. $\hat{\boldsymbol{f}}$ and $\hat{\boldsymbol{\tau}}$ are chosen such that they satisfy the bounds on the forces and torques defined in (22). Once the spacecraft reaches $\boldsymbol{p}^{current}$, it decelerates under a constant force -$\hat{\boldsymbol{f}}$ and a constant torque -$\hat{\boldsymbol{\tau}}$ until it stops at the next node $\boldsymbol{p}^{next}$. This is simple way to guarantee that the controls of the spacecraft are satisfied along each consecutive nodes. The smaller $\hat{\boldsymbol{f}}$ and $\hat{\boldsymbol{\tau}}$ 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 $\boldsymbol{p}^{inter}$ 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_{max} is computed. t_{max} represents the maximum time needed by all spacecraft to reach $\boldsymbol{p}^{inter}$, if they all move under the same constant force $\hat{\boldsymbol{f}}$ and rotate under the same constant torque $\hat{\boldsymbol{\tau}}$. Then fixing the time to reach $\boldsymbol{p}^{inter}$ to be t_{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 $\hat{\boldsymbol{f}}$, and torques smaller than $\hat{\boldsymbol{\tau}}$. Algorithm 3 contains the main steps of the *transition* algorithm. $traj$ is the RRT output of the first stage, and $\boldsymbol{p}^k$ represents a point in the composite trajectories of all the spacecraft at node k (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 updated nodes. Lines 2-4 in Algorithm 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 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 5 for the steps involved in GET-MAX-TIME. Continuing with Algorithm 4, line 3 updates the forces and torques of the current node $\boldsymbol{p}^{current}$, which are ensured to be feasible by construction. The velocities at $\boldsymbol{p}^{current}$ remain zeros to satisfy the assumption of the first stage algorithm. Line 4 propagates the dynamics to the intermediate node $\boldsymbol{p}^{inter}$ by using the logic explained earlier in this section.

Algorithm 3 AUGMENT-TRAJECTORY($traj$)

1: $augmented\text{-}traj \leftarrow \emptyset$
2: **for** $k \leftarrow 1$ to SIZE($traj$)-1 **do**
3: $\quad augmented\text{-}traj \leftarrow$ APPEND($augmented\text{-}traj$, PROPAGATE-DYNAMICS($\boldsymbol{p}^k$, $\boldsymbol{p}^{k+1}$))
4: **end for**
5: $augmented\text{-}traj \leftarrow$ APPEND($augmented\text{-}traj$, UPDATE-NODE($\boldsymbol{p}^{end}$))
6: **return** $augmented\text{-}traj$

Algorithm 4 PROPAGATE-DYNAMICS($\boldsymbol{p}^{current}$, $\boldsymbol{p}^{next}$)

1: $\boldsymbol{t}_{max} \leftarrow$ GET-MAX-TIME($\boldsymbol{p}^{current}$, $\boldsymbol{p}^{next}$)
2: **for** $i \leftarrow 1$ to N **do**
3: $\quad \boldsymbol{p}_i^{current} \leftarrow$ UPDATE-NODE($\boldsymbol{p}_i^{current}$, $\boldsymbol{p}_i^{next}$, $\boldsymbol{t}_{max}$)
4: $\quad \boldsymbol{p}_i^{inter} \leftarrow$ PROP-CURRENT-DYN($\boldsymbol{p}_i^{current}$, $\boldsymbol{t}_{max}$)
5: **end for**
6: **return** $\{\boldsymbol{p}^{current}, \boldsymbol{p}^{inter}\}$

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 5 GET-MAX-TIME($\boldsymbol{p}^{current}$, $\boldsymbol{p}^{next}$)

1: $\boldsymbol{t}_{max} \leftarrow 0$
2: **for** $i \leftarrow 1$ to N **do**
3: $\quad \boldsymbol{t}_{max} \leftarrow$ MAX(t_{max}, GET-MAX-ROTATION-TIME($\boldsymbol{p}_i^{current}$, $\boldsymbol{p}_i^{next}$))
4: $\quad \boldsymbol{t}_{max} \leftarrow$ MAX(t_{max}, GET-MAX-TRANSLATION-TIME($\boldsymbol{p}_i^{current}$, $\boldsymbol{p}_i^{next}$))
5: **end for**
6: **return** t_{max}

C. 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 [44]. The augmented RRT output of the first stage is used as initial guess in solving the NLP, and is essential in 1) reducing the computation times of the solution process. and 2) solving more complex reconfiguration problems.

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 [45]. 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" [46].

This paper uses a Gauss pseudospectral method, which has been shown to have, in general, more accurate solutions than other pseudospectral methods [38]. 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) [47]. 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. [38] and [47] 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 [48], [47]. Determine the state, $\boldsymbol{x}$(t), and control, $\boldsymbol{u}$(t), that minimize the cost functional

$$J = \Phi(\boldsymbol{x}(t_0), t_0, \boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} g(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt \quad (25)$$

subject to the dynamic constraints

$$\dot{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \in \mathbb{R}^n \quad (26)$$

the boundary condition

$$\boldsymbol{\phi}(\boldsymbol{x}(t_0), t_0, \boldsymbol{x}(t_f), t_f) = \boldsymbol{0} \in \mathbb{R}^q \quad (27)$$

the inequality path constraints

$$\boldsymbol{C}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \leq \boldsymbol{0} \in \mathbb{R}^c \quad (28)$$

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 (25)-(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

$$\varsigma = \frac{2t}{t_f - t_0} - \frac{t_f + t_0}{t_f - t_0} \quad (29)$$

Rewrite the optimal control problem after replacing t with ς:

$$J = \Phi(\boldsymbol{x}(-1), t_0, \boldsymbol{x}(1), t_f) + \frac{t_f - t_0}{2} \int_{-1}^{1} g(\boldsymbol{x}(\varsigma), \boldsymbol{u}(\varsigma), \varsigma; t_0, t_f)d\varsigma \quad (30)$$

subject to the constraints

$$\dot{\boldsymbol{x}} = \frac{t_f - t_0}{2} \boldsymbol{F}(\boldsymbol{x}(\varsigma), \boldsymbol{u}(\varsigma), \varsigma; t_0, t_f) \in \mathbb{R}^n \quad (31)$$

$$\boldsymbol{\phi}(\boldsymbol{x}(-1), t_0, \boldsymbol{x}(1), t_f) = \boldsymbol{0} \in \mathbb{R}^q \quad (32)$$

$$\boldsymbol{C}(\boldsymbol{x}(\varsigma), \boldsymbol{u}(\varsigma), \varsigma; t_0, t_f) \leq \boldsymbol{0} \in \mathbb{R}^c \quad (33)$$

where $\varsigma \in [-1, 1]$. (30)-(33) is called the transformed continuous Bolza problem.

In the GPM, the set of $\mathcal{N}$ discretization points includes $\mathcal{K} = \mathcal{N} - 2$ 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 $\mathcal{K}$+1 Lagrange interpolating polynomials, $\mathcal{L}_i$, $i = 0 \ldots \mathcal{K}$,

$$\boldsymbol{x}(\varsigma) \approx \boldsymbol{X}(\varsigma) = \sum_{i=0}^{\mathcal{K}} \boldsymbol{X}(\varsigma_i)\mathcal{L}_i(\varsigma) \quad (34)$$

where

$$\mathcal{L}_i(\varsigma) = \prod_{j=0, j\neq i}^{\mathcal{K}} \frac{\varsigma - \varsigma_i}{\varsigma_i - \varsigma_j} \quad (35)$$

The control is approximated using a basis of $\mathcal{K}$ Lagrange interpolating polynomials $\mathcal{L}_i^{\dagger}$, $i = 1 \ldots \mathcal{K}$

$$\boldsymbol{u}(\varsigma) \approx \boldsymbol{U}(\varsigma) = \sum_{i=1}^{\mathcal{K}} \boldsymbol{U}(\varsigma_i)\mathcal{L}_i^{\dagger}(\varsigma) \quad (36)$$

where

$$\mathcal{L}_i^{\dagger}(\varsigma) = \prod_{j=1, j\neq i}^{\mathcal{K}} \frac{\varsigma - \varsigma_i}{\varsigma_i - \varsigma_j} \quad (37)$$

The dynamic constraints are transcribed into algebraic constraints as follows

$$\sum_{i=0}^{\mathcal{K}} D_{ki}\boldsymbol{X}_i - \frac{t_f - t_0}{2}\boldsymbol{F}(\boldsymbol{X}(\varsigma_k), \boldsymbol{U}(\varsigma_k), \varsigma_k; t_0, t_f) = \mathbf{0} \tag{38}$$

where $k = 1 \dots \mathcal{K}$, and D is an $\mathcal{K} \times (\mathcal{K}+1)$ 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:

$$D_{ki} = \dot{\mathcal{L}}_i(\varsigma_k) = \sum_{l=0}^{\mathcal{K}} \frac{\prod_{j=0, j\neq i, l}^{\mathcal{K}} (\varsigma_k - \varsigma_j)}{\prod_{j=0, j\neq i}^{\mathcal{K}} (\varsigma_i - \varsigma_j)} \tag{39}$$

where $k = 1 \dots \mathcal{K}$ and $i = 0 \dots \mathcal{K}$. Note that the collocation of the dynamic constraint only happens at the LG points and not at the boundary points. Two additional variables, $\boldsymbol{X}_0$ and $\boldsymbol{X}_f$ are defined in this discretization. $\boldsymbol{X}_0 \equiv \boldsymbol{X}(-1)$, and $\boldsymbol{X}_f$ is defined via a Gauss quadrature

$$\boldsymbol{X}_f \equiv \boldsymbol{X}_0 + \frac{t_f - t_0}{2}\sum_{k=1}^{\mathcal{K}} w_k \boldsymbol{F}(\boldsymbol{X}(\varsigma_k), \boldsymbol{U}(\varsigma_k), \varsigma_k; t_0, t_f) \tag{40}$$

where w_k are the Gauss weights.

Continuing with the transcription process, (30) is approximated using a Gauss quadrature

$$\begin{aligned} J = {} & \Phi(\boldsymbol{X}(-1), t_0, \boldsymbol{X}(1), t_f) \\ & + \frac{t_f - t_0}{2}\sum_{k=1}^{\mathcal{K}} w_k g(\boldsymbol{X}(\varsigma_k), \boldsymbol{U}(\varsigma_k), \varsigma_k; t_0, t_f) \end{aligned} \tag{41}$$

The boundary constraint is written as

$$\boldsymbol{\phi}(\boldsymbol{X}(-1), t_0, \boldsymbol{X}(1), t_f) = \mathbf{0} \tag{42}$$

Finally, the path constraint is computed at the LG points as

$$\boldsymbol{C}(\boldsymbol{X}(\varsigma_k), \boldsymbol{U}(\varsigma_k), \varsigma_k; t_0, t_f) \leq \mathbf{0} \tag{43}$$

where $k = 1 \dots \mathcal{K}$. Equations (38), (40), (41), (42) and (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.

VI. 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, and the dimensions of the test environment are similar to those of the SPHERES testbed on ISS [49]. Note that some of the examples are motivated by reconfiguration maneuvers described in Ref. [9].

A. 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 [50]. GPOCS is a MATLAB implementation of the Gauss pseudospectral method for solving optimal control problems. GPOCS relies on SNOPT [44], 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 $\mathcal{N} = 25$.

B. 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 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 computation times are 2 sec for the first stage, and 65 sec for the second stage. Figure 3 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 4 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.

C. 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 $[0.5, 0.7, 0.5]^T$

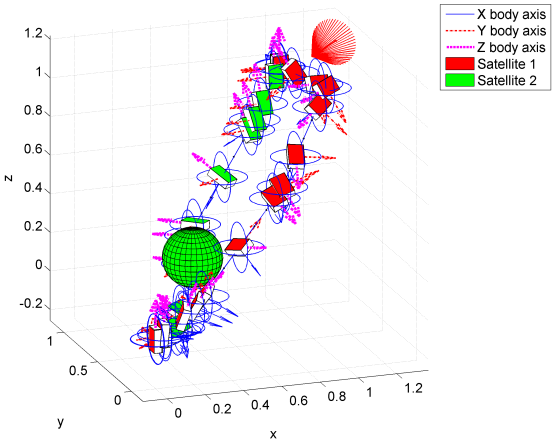


Fig. 3. Example: Two-Spacecraft Maneuver with Obstacle and Sun Avoidance. RRT Output. Spacecraft 1 and 2 switch positions while avoiding colliding with a fixed obstacle and pointing to the sun.

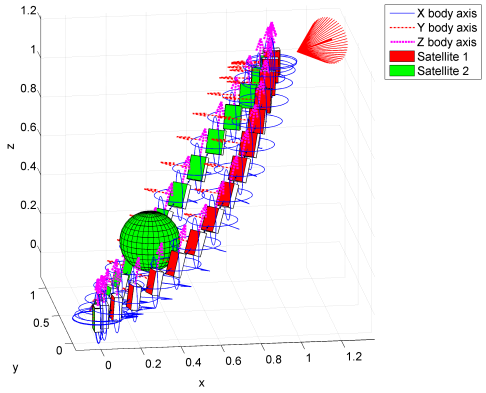


Fig. 4. Final trajectory of the two-spacecraft maneuver with obstacle and sun avoidance example.

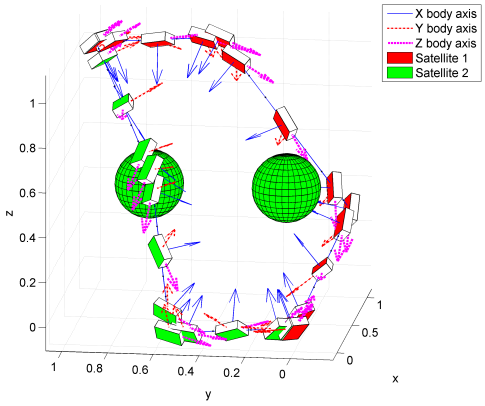


Fig. 5. Example: Two-Spacecraft Maneuver with Inter-Spacecraft Pointing and Obstacle Avoidance. RRT Output. Spacecraft 1 and 2 switch positions while avoiding colliding with two fixed obstacles and keep pointing to each other within 32°.

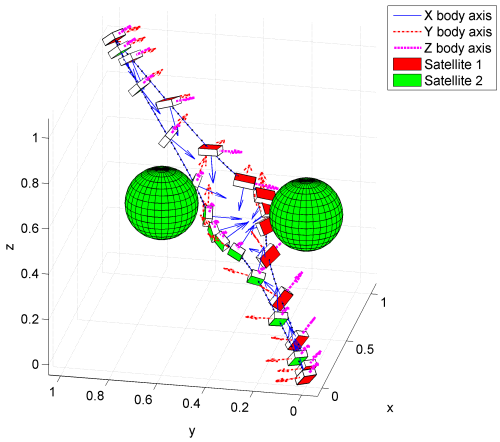


Fig. 6. Final trajectory of the two-spacecraft maneuver with obstacle and inter-spacecraft pointing.

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 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 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 9 shows the inter-spacecraft collision avoidance constraint which is active for a large part of the maneuver, when the spacecraft enter the narrow passage between the obstacles. Figures 7 and 8 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 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 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.

D. 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°. Spacecraft 1 and 2 are initially

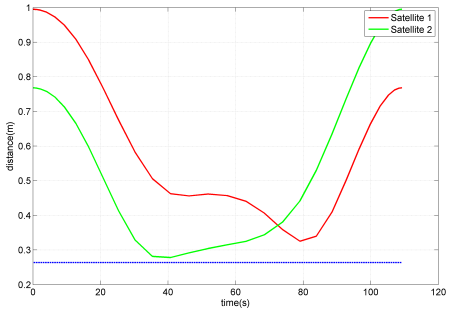


Fig. 7. Distances between centers of the spacecraft and center of obstacle located at $[0.5, 0.7, 0.5]^T$ for the coupled two-spacecraft maneuver.

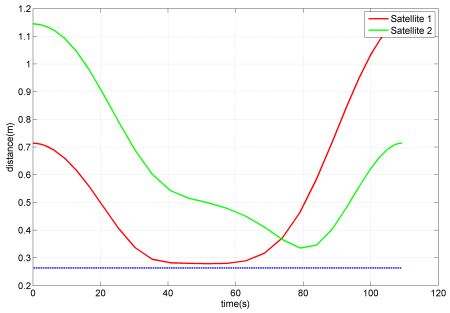


Fig. 8. Distances between centers of the spacecraft and center of obstacle located at $[0.5, 0.1, 0.5]^T$ for the coupled two-spacecraft maneuver.

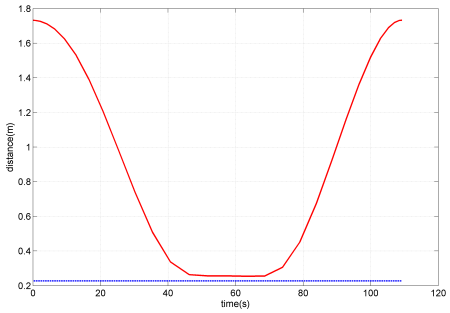


Fig. 9. Distance between the centers of the spacecraft for the coupled two-spacecraft maneuver. The dashed line shows the minimum permissible distance

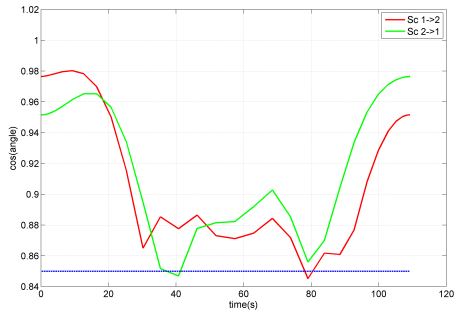


Fig. 10. Cosines of angles between ranging device vector and relative position of both spacecraft for the coupled two-spacecraft maneuver.

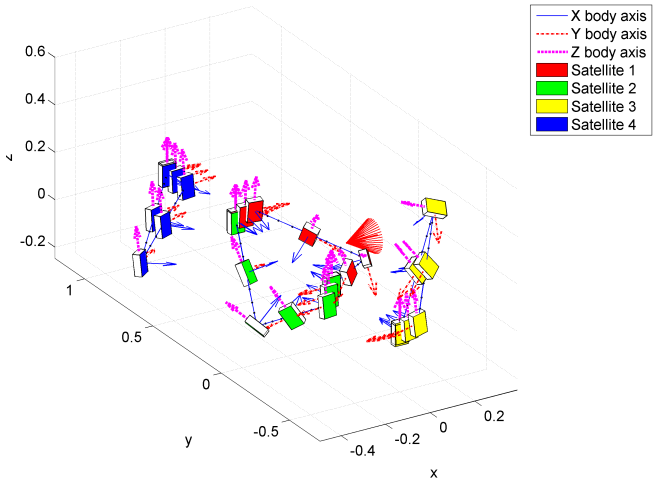


Fig. 11. Example: Four-Spacecraft Maneuver with Inter-Spacecraft Pointing and Absolute Pointing. RRT Output. Spacecraft 1 and 2 switch positions while keep pointing to each other within 33°. Spacecraft 3 and 4 keep both spacecraft 1 and 2 in their specified cone of 30°.

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°. All four spacecraft must also avoid pointing their X body axis in the sun cone. The sun cone is represented by the vector $[1, 0, 0]^T$ pointing at the sun and surrounded by a 20° half angle cone. 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 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 12 shows the final trajectory produced by the RRT-GPM planner. It is a smoothed version of Figure 11. Figure 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.

The following section is another contribution of this paper. It underlines 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 an RRT technique as a way to initialize pseudospectral methods. Finally, it shows the improvement that the RRT initialization brings to spacecraft

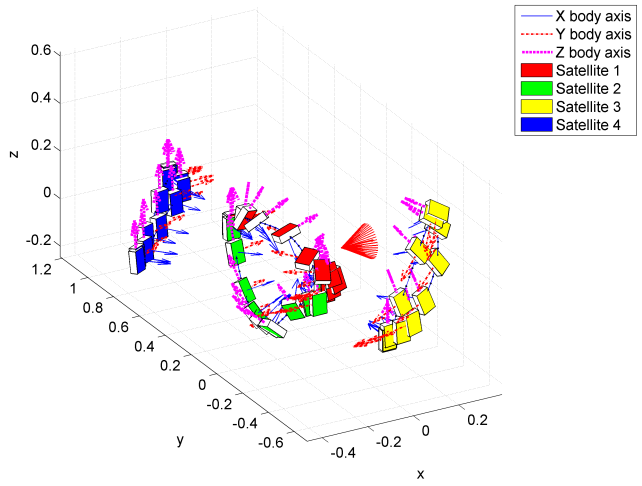


Fig. 12. Final trajectory of the four-spacecraft maneuver with inter-spacecraft and absolute pointing.

reconfiguration problems, which are solved using a Gauss pseudospectral method.

VII. 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 usually 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 [51]), 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 [44]).

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 [52]. 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 [47]. 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 [47]. 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.

A. Survey of Initialization Techniques

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 [53]. 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 [26]. 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 [44], [54]. 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 [55]. 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

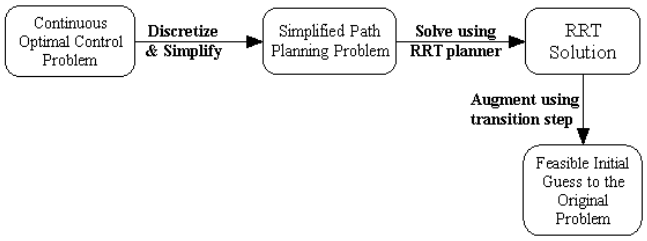


Fig. 13. Steps Involved in Generating a Feasible Initial Guess to the Optimal Control Problem. This guess serves as a starting point to the NLP resulting from the transcription of the optimal control problem using Pseudospectral methods.

version of a path planning problem without differential constraints.

B. RRT Planner Initialization

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. Refer to Section III-B for more details about RRTs. 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 paper. The resulting output is a feasible initial guess to the complete optimal control problem.

The paper uses an improved version of the bidirectional rapidly-exploring random trees (RRT) that is described in Ref. [21]. The *transition* phase is developed in Section V-B. The bidirectional 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 13.

C. Illustration of the RRT improvement

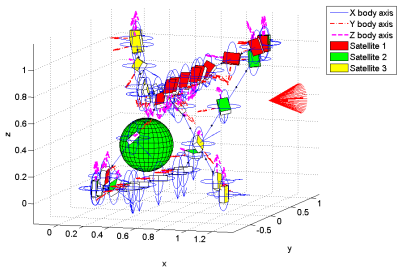
This section illustrates the improvement that the RRT initialization step brings to the solution of an optimal control problem transcribed into an NLP using a recently developed pseudospectral method method called the Gauss pseudospectral method (GPM) (Section V-C). This method has shown promise both in the accuracy of the solution and post-optimality analysis of optimal control problems [39]. The problem is a multispacecraft reconfiguration maneuver with path constraints that is described in Section IV. 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 *i.e.,* following the two-stage approach described in Section V, and 2) using a "cold" start approach which leaves it to the nonlinear solver to find an initial starting guess. Refer to Section VI for details related to the figures illustrating reconfiguration maneuver examples.

The examples consist of reconfiguration maneuvers that include stay outside constraints (*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 VI-C and Section VI-D. 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 *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 [50]. Both optimality and feasibility tolerances are set to 10^{-4}. For more details about the implementation, refer to Section VI-A. The computation times and costs of the following examples are summarized in Table I and Table II.

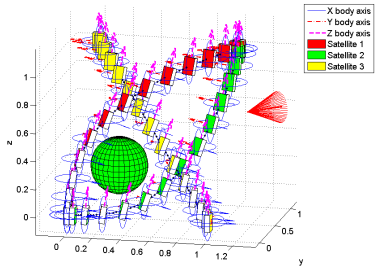
1) Single Spacecraft Maneuver: This example is a simple translation from the position $[0,0,0]^T$ to $[1,1,1]^T$ with a 180° rotation around the Z axis. The spacecraft has to avoid pointing its X axis in the sun direction, which is represented by the vector $[\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}},0]^T$, 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.

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 VII-C.1 applies in this problem. Figure 14 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.

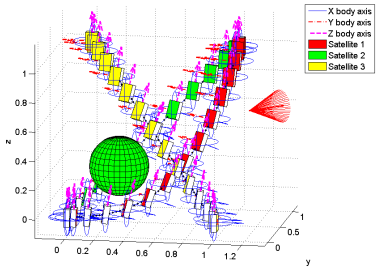
3) Formation Reflection with Four Spacecraft: This example consists of four spacecraft that start in a square formation, and end in another reflected square formation. Spacecraft 1 and spacecraft 3 start at $[0,1,0]^T$ and $[2,1,0]^T$, respectively. They must end at their original positions. They must 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,



(a) RRT Initial Guess

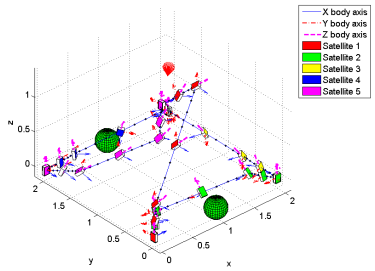


(b) Final trajectory solved using RRT initial guess.

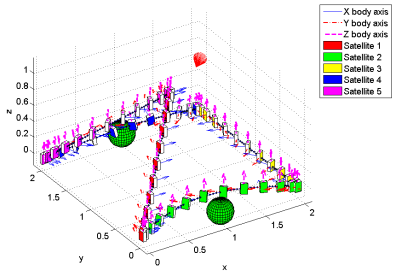


(c) Final trajectory solved with a cold start approach.

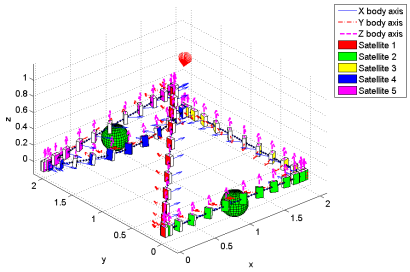
Fig. 14. RRT Path and Final Trajectories for the Three-Spacecraft Example



(a) RRT Initial Guess



(b) Final trajectory solved using RRT initial guess.



(c) Final trajectory solved with a cold start approach.

Fig. 15. RRT Path and Final Trajectories for the Five-Spacecraft Example

the maneuver is a reflection of a square formation around a line passing through the fixed positions of Spacecraft 1 and 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 $[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0]^T$, surrounded by a cone of 20° half angle. They must also avoid colliding with two fixed obstacles of radius 0.15 m. Figure 15 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.

D. Performance Comparison

Table I 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.

TABLE I. COMPARISON OF COMPUTATION TIMES OF RECONFIGURATION MANEUVERS FOR FORMATION OF INCREASING SIZE SOLVED USING A GAUSS PSEUDOSPECTRAL METHOD (AVERAGE OVER 10 RUNS)

Example	Time (s) w/o RRT	Time (s) with RRT	Time Ratio
1 s/c (VII-C.1)	35	24	1.46
2 s/c (VI-B)	103	67	1.54
3 s/c (VII-C.2)	335	171	1.96
4 s/c (VII-C.3)	478	228	2.10
5 s/c (VII-C.4)	834	356	2.34

Figure 16 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

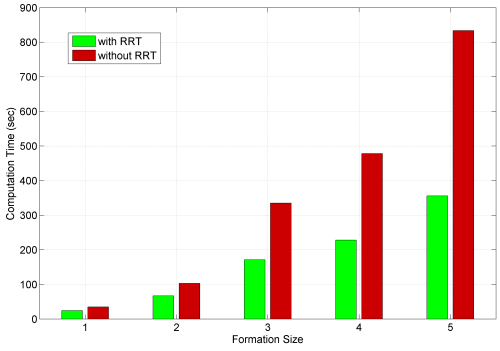


Fig. 16. Comparison of computation times of a series of reconfiguration maneuvers of increasing size solved using a Gauss pseudospectral method (average over 10 runs). The *with RRT* includes computation times of the RRT planner and the *transition* step.

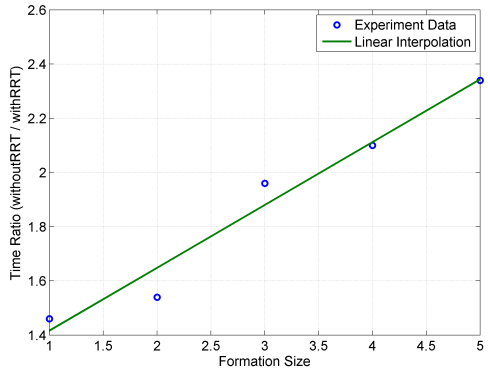


Fig. 17. Time Ratios of *w/o RRT* over *with RRT* Computation Times. The *with RRT* includes computation times of the RRT planner and the *transition* step.

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 17 illustrates this increase in time ratio. The time ratio values can be fit by a straight line with equation

$$TimeRatio(N) = 0.23N + 1.18 \tag{44}$$

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

$$without\ RRT\ Time(N) \approx 31.4N^{2.01} \tag{45}$$

$$with\ RRT\ Time(N) \approx 23.3N^{1.69} \tag{46}$$

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 II. COMPARISON OF FINAL COSTS OF RECONFIGURATION MANEUVERS FOR FORMATION OF INCREASING SIZE SOLVED USING A GAUSS PSEUDOSPECTRAL METHOD (AVERAGE OVER 10 RUNS)

Example	Cost w/o RRT	Cost with RRT	Cost Comparison %
1 s/c (VII-C.1)	0.975	1.012	+3.65
2 s/c (VI-B)	1.806	1.847	+2.22
3 s/c (VII-C.2)	2.752	2.821	+2.45
4 s/c (VII-C.3)	4.490	4.444	−1.03
5 s/c (VII-C.4)	8.654	8.743	+1.02

Table II 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 percentage of improvement (*i.e.,* decrease) in the cost of the *w/o RRT* case when compared to the *with RRT* one. One would expect that the *w/o RRT* costs to be better than those of the *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 *w/o RRT* version is 3.65% better. For the largest formation, the five spacecraft reconfiguration described in Section VII-C.4, the *w/o RRT* cost is only 1.02% better.

Finally, it should be emphasized that GPOCS fails to solve examples described in Sections VI-C and VI-D after 10 attempts, when not initialized with an RRT initial guess. These examples include coupled constraints and they are therefore very challenging for the SNOPT nonlinear solver that GPOCS relies on in its solution process. However, when provided with an RRT guess, GPOCS is able to converge to an optimal solution. So not only does the RRT guess reduce the computation times of the reconfiguration maneuver problem, it also enables the solution of a complex class of reconfiguration problems that include inter-spacecraft constraints.

VIII. CONCLUSIONS

This paper presented a two-stage path planning technique for designing multiple spacecraft reconfiguration maneuvers with various path constraints. The primary idea was to combine an improved RRT planner with a Gauss pseudospectral method to obtain highly accurate solutions in computation times reasonable for online implementation. Several complex examples demonstrated the validity of this approach. This paper also showed the importance of an initialization step based on an RRT planner to the solution of problems solved using pseudospectral methods. It showed that this step 1) reduced the computation time of the solutions, and 2) made it possible to solve a more complex class of such problems.

REFERENCES

[1] P. R. Lawson, "The Terrestrial Planet Finder," in *Proceedings of the IEEE Aerospace Conference*, vol. 4, March 2001, pp. 2005–2011.

[2] "NASA Laser Interferometer Space Antenna Website." [Online]. Available: http://lisa.nasa.gov/

[3] W. Cash, N. White, and M. Joy, "The Maxim Pathfinder Mission - X-ray imaging at 100 micro-arcseconds," in *Proceedings of the Meeting on X-ray optics, instruments, and missions III*, vol. 4012, Munich, Germany, 2000, pp. 258–269.

[4] "Broad Agency Announcement (BAA07-31) System F6 for Defense Advanced Research Projects Agency (DARPA)," 2007. [Online]. Available: http://www.darpa.mil/ucar/solicit/baa0731/f6_baa_final_07-16-07.doc

[5] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen, "A Survey of Spacecraft Formation Flying Guidance and Control. Part II: Control," in *Proceedings of the American Control Conference*, vol. 4, 30 June-2 July 2004, pp. 2976–2985.

[6] "NASA, the Terrestrial Planet Finder mission." [Online]. Available: http://planetquest.jpl.nasa.gov/TPF/tpf_index.cfm

[7] G. Inalhan, M. Tillerson, and J. P. How, "Relative Dynamics and Control of Spacecraft Formations in Eccentric Orbits," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 48–59, January 2002.

[8] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen, "A Survey of Spacecraft Formation Flying Guidance and Control. Part I: Guidance," in *Proceedings of American Control Conference*, vol. 2, Jun 2003, pp. 1733–1739.

[9] I. M. Garcia, "Nonlinear Trajectory Optimization with Path Constraints Applied to Spacecraft Reconfiguration Maneuvers," Master's thesis, Massachussets Institute of Technology, 2005.

[10] C. Sultan, S. Seereram, and R. K. Mehra, "Deep Space Formation Flying Spacecraft Path Planning," *The International Journal of Robotics Research*, vol. 26, no. 4, pp. 405–430, 2007.

[11] A. Richards, T. Schouwenaars, J. P. How, and E. Feron, "Spacecraft Trajectory Planning with Avoidance Constraints," *Journal of Guidance control and dynamics*, vol. 25, no. 4, pp. 755–764, 2002.

[12] R. K. Prasanth, J. D. Boskovic, and R. K. Mehra, "Mixed integer/LMI programs for low-level path planning," in *Proceedings of the American Control Conference*, vol. 1, 2002, pp. 608–613 vol.1.

[13] F. McQuade, R. Ward, and C. McInnes, "The Autonomous Configuration of Satellite Formations using Generic Potential Functions," in *Proceedings of the International Symposium Formation Flying*, 2002.

[14] R. O. Saber, W. B. Dunbar, and R. M. Murray, "Cooperative Control of Multi-Vehicle Systems using Cost Graphs and Optimization," in *Proceedings of the American Control Conference*, vol. 3, 2003, pp. 2217–2222.

[15] E. Frazzoli, M. A. Dahleh, E. Feron, and R. P. Kornfeld, "A Randomized Attitude Slew Planning Algorithm for Autonomous Spacecraft," *AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada*, 2001.

[16] L. E. Kavraki, P. Svestka, C. L. J, and M. H. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[17] L. E. Kavraki, M. N. Kolountzakis, and J. C. Latombe, "Analysis of probabilistic roadmaps for path planning," in *IEEE International Conference on Robotics and Automation*, vol. 4, 1996, pp. 3020–3025 vol.4.

[18] J. M. Phillips, L. E. Kavraki, and N. Bedrosian, "Spacecraft Rendezvous and Docking With Real-Time Randomized Optimization," in *AIAA Guidance, Navigation and Control Conference*, Austin, TX, August 2003.

[19] E. Frazzoli, "Quasi-Random Algorithms for Real-Time Spacecraft Motion Planning and Coordination," *Acta Astronautica*, vol. 53, no. 4-10, pp. 485–495, 2003.

[20] I. Garcia and J. P. How, "Trajectory Optimization for Satellite Reconfiguration Maneuvers With Position and Attitude Constraints," in *American Control Conference, 2005. Proceedings of the 2005*, vol. 2, 2005, pp. 889–894.

[21] ——, "Improving the Efficiency of Rapidly-exploring Random Trees Using a Potential Function Planner," in *IEEE Conference on Decision and European Control Conference*, 2005, pp. 7965–7970.

[22] G. T. Huntington and A. V. Rao, "Optimal Configuration Of Spacecraft Formations Via A Gauss Pseudospectral Method," *Advances in the Astronautical Sciences, Part I*, vol. 120, pp. 33–50, 2005.

[23] Q. Gong, W. Kang, and I. M. Ross, "A Pseudospectral Method for the Optimal Control of Constrained Feedback Linearizable Systems," in *Proceedings of Conference on Control Applications on*, 2005, pp. 1033–1038.

[24] I. Ross and C. D'Souza, "Rapid Trajectory Optimization of Multi-Agent Hybrid Systems," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004.

[25] J. H. Reif, "Complexity of the Mover's Problem and Generalizations," in *IEEE Symposium on Foundations of Computer Science*, 1979, pp. 421–427.

[26] J. T. Betts and W. P. Huffman, "Mesh Refinement in Direct Transcription Methods for Optimal Control," *Optimal Control Applications and Methods*, vol. 19, no. 1, pp. 1–21, 1998.

[27] B. R. Donald, P. G. X. J. F. Canny, and J. H. Reif, "Kinodynamic Motion Planning," *Journal of the ACM*, vol. 40, no. 5, pp. 1048–1066, 1993.

[28] J. J. Kuffner, "Motion Planning with Dynamics," Physiqual, March 1998.

[29] D. Hsu, R. Kindel, J. Latombe, and S. Rock, "Randomized Kinodynamic Motion Planning with Moving Obstacles," *Int. J. Robotics Research*, vol. 21, no. 3, pp. 233–255, 2002.

[30] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time Motion Planning for Agile Autonomous Vehicles," in *American Control Conference*, vol. 1, 2001, pp. 43–49.

[31] I. Belousov, C. Esteves, J. P. Laumond, and E. Ferre, "Motion Planning for the Large Space Manipulators with Complicated Dynamics," in *International Conference on Intelligent Robots and Systems, IEEE/RSJ*, 2005, pp. 2160–2166.

[32] S. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning. Technical Report 98-11, Computer Science Dept., Iowa State University," Oct 1998.

[33] S. M. LaValle and J. J. J. Kuffner, "Randomized kinodynamic planning," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 1, 1999, pp. 473–479.

[34] S. M. Lavalle, *Planning Algorithms*. Cambridge University Press, 2006.

[35] J. T. Betts, "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[36] G. Elnagar, M. A. Kazemi, and M. Razzaghi, "The pseudospectral Legendre Method for Discretizing Optimal Control Problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 10, pp. 1793–1796, 1995.

[37] G. N. Elnagar and M. A. Kazemi, "Pseudospectral Chebyshev Optimal Control of Constrained Nonlinear Dynamical Systems," *Comput. Optim. Appl.*, vol. 11, no. 2, pp. 195–217, 1998.

[38] D. Benson, "A Gauss Pseudospectreal Transcription for Optimal Control," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[39] G. T. Huntington and A. V. Rao, "Optimal Reconfiguration Of A Tetrahedral Formation Via A Gauss Pseudospectral Method," *Advances in the Astronautical Sciences, Part II*, vol. 123, pp. 1337–1358, 2006.

[40] M. D. Shuster, "Survey of Attitude Representations," *Journal of the Astronautical Sciences*, vol. 41, pp. 439–517, Oct. 1993.

[41] A. G. Richards, "Trajectory Optimization using Mixed-Integer Linear Programming," Master's thesis, Massachusetts Institute of Technology, June 2002.

[42] Y. Kim, M. Mesbah, and F. Y. Hadaegh, "Dual-spacecraft formation flying in deep space - Optimal collision-free reconfigurations," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 375–379, March 2003.

[43] A. Rahmani, M. Mesbahi, and F. Y. Hadaegh, "On the Optimal Balanced-Energy Formation Flying Maneuvers," *2005 AIAA Guidance, Navigation, and Control Conference and Exhibit; San Francisco, CA; USA; 15-18 Aug. 2005*, pp. 1–8, 2005.

[44] P. E. Gill, W. Murray, and M. A. Saunders, "User's Guide For SNOPT 5.3: A Fortran Package For Large-Scale Nonlinear Programming," 1999.

[45] I. M. Ross and F. Fahroo, "Convergence of Pseudospectral Discretizations of Optimal Control Problems," in *Proceedings of the 40th IEEE Conference on Decision and Control*, vol. 4, 2001, pp. 3175–3177.

[46] L. N. Trefethen, *Spectral Methods in Matlab*. Philadelphia: SIAM Press, 2000.

[47] G. T. Huntington, "Advancement and Analysis of a Gauss Pseudospectral Transcription for Optimal Control Problems," Ph.D. dissertation, Massachusetts Institute of Technology, June 2007.

[48] D. E. Kirk, *Optimal Control Theory: An Introduction*. Dover Publications, 1970.

[49] A. S. Otero, A. Chen, D. W. Miller, and M. Hilstad, "SPHERES: Development of an ISS Laboratory for Formation Flight and Docking Research," in *IEEE Aerospace Conference Proceedings*, vol. 1, 2002, pp. 59–73.

[50] A. V. Rao, *User's Manual for GPOCS 1 beta: A MATLAB Implementation of the Gauss Pseudospectral Method for Solving Non-Sequential Multiple-Phase Optimal Control Problems*, report 1 beta ed., May 2007.

[51] R. J. Vanderbei and D. F. Shanno, "An Interior Point Algorithm for Nonconvex Nonlinear Programming," *Computational Optimization and Applications*, vol. 13, pp. 231–252, 1999.

[52] N. Muscettola, B. Smith, S. Chien, C. Fry, G. Rabideau, K. Rajan, and D. Yan, "On-board Planning for Autonomous Spacecraft," in *In Proceedings of the Fourth International Symposium on Artificial Intelligence, Robotics, and Automation for Space (iSAIRAS)*, July 1997.

[53] I. M. Ross, "User's Manual for DIDO: A MATLAB Application Package for Solving Optimal Control Problems," Monterey, California, Technical Report 04-01.0, February 2004.

[54] J. Mitchell and B. Borchers, "Solving Real-World Linear Ordering Problems Using a Primal-Dual Interior Point Cutting Plane Method," *Annals of Operations Research*, vol. 62, no. 1, pp. 253–276, Dec. 1996.

[55] R. A. Bartlett, A. Wachter, and L. T. Biegler, "Active Set vs. Interior Point Strategies for Model Predictive Control," in *Proceedings of the American Control Conference*, vol. 6, 2000, pp. 4229–4233.