JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 28, No. 4, July–August 2005

Hybrid Optimal Control Framework for Mission Planning

I. Michael Ross*
Naval Postgraduate School, Monterey, California 93943
and
Christopher N. D'Souza†
Charles Stark Draper Laboratory, Inc., Houston, Texas 77058

With the progressive sophistication of future missions, it has become increasingly apparent that a new framework is necessary for efficient planning, analysis, and optimization of various concepts of operations (CONOPS). In recognizing that CONOPS involve categorical variables, we propose a hybrid optimal control framework that mathematically formalizes such problems. Hybrid optimal control theory extends ordinary optimal control theory by including categorical variables in the problem formulation. The proposed formalism frees mission planners to focus on high-level decision making by automating and optimizing the details of the inner loops. The eventual goal of this formalism is to develop efficient tools and techniques to support the objective of increasing autonomy for future systems. In using the pseudospectral knotting method to solve hybrid optimal control problems, we generate a mixed-variable programming (MVP) problem. A simple, feasible integer programming subproblem is identified that reduces the combinatorial complexity of solving the MVP. In addition to developing the framework using various examples from aerospace engineering, we provide details for a two-agent benchmark problem associated with a multiagent launch system. The entire process is illustrated with a numerical example.

I. Introduction

WITH the increasing sophistication of future missions, there is a growing need to manage complexity efficiently. The need to manage complexity imposes new challenges to designing autonomous systems. Future challenges in autonomous systems are skewed more toward the design of the outer loops (Fig. 1) rather than the inner loop. This is at least in part because it is now possible to solve routinely[1–6] a wide class of optimal control problems and some in real time as well.[6–8] The success of inner-loop control has emboldened mission planners to conceive of missions of such enormous complexity that only high-level decision making, consisting of simple, intuitive, linguistic commands, is expected to be allocated to an operator. The natural progression of the architecture, shown in Fig. 1, is to eliminate the operator from the loop, in which case we regard the autonomous system as "intelligent." Although the details of an intelligent system are not shown in Fig. 1, this architecture, and the development of the hybrid optimal control framework, are best illustrated by a few examples.

A. Example 1, Asteroid Mission

Given a database of asteroids of interest, design a sample-return, low-thrust space mission that selects and maximizes the number of asteroids visited while minimizing fuel consumption.

Ignoring the many details of this problem, such as the desired duration of the visit for a given asteroid, it is immediately apparent that this is not a standard low-thrust trajectory optimization problem because the type of decision variables are continuous (thrust) as well as discrete (names/number of asteroids). Hence, to maintain a direct connection to linguistic information, we refer to the totality of discrete variables as categorical variables.[9,10] A formal way to tackle this problem is to combine optimal control theory of differential equations with that of discrete event systems. This combination can be visualized in the form of a directed graph (digraph), as shown in Fig. 2. See Ref. 11 for a quick introduction to graph theory, and the Stateflow® manual[12] for a practical means to model and simulate such systems. In Fig. 2, the universe consists of just two asteroids in addition to the Earth and the sun. Each vertex q of the digraph represents a continuous-time controlled dynamics of the spacecraft with respect to the celestial body of interest, whereas the edges represent a switch of the spacecraft's dynamics from one vector field to another. The absence of an edge between q_a and q_b implies that, for the spacecraft to transition from orbiting about asteroid A to orbiting about asteroid B, it must necessarily transition first to a heliocentric motion q_s. Note that the vertices in Fig. 2 do not represent the asteroids as in the case of cities either in the standard traveling salesman problem[13] or the motorized traveling salesman problem proposed by von Stryk (see Refs. 14–16).

A segment of the mission plan that calls for the spacecraft to go from asteroid A to asteroid B can be represented by the discrete event sequence

$$\boldsymbol{q}_{AB} = (q_a, q_s, q_b)$$

The loop about the vertex q_e in Fig. 2 represents a possible jump discontinuity (called a reset map, discussed Sec. II) in the mass of the spacecraft as a result of dropping a sample to Earth, or of dropping the upper stage of the rocket during the initial burn. The categorical state space $\mathcal{Q}$ for the system shown in Fig. 2 is the finite set $\mathcal{Q} = \{q_a, q_b, q_s, q_e\}$ of cardinality, $N_Q = 4$. Thus, a high-level mission plan can be encoded as a walk[11] from one vertex to another; for example, the sequence

$$\boldsymbol{q}_M = (q_e, q_s, q_a, q_s, q_e, q_e, q_s, q_b)$$

corresponds to a low-thrust escape from Earth orbit, followed by a minimum-fuel heliocentric trajectory, an orbit about asteroid A and a return to Earth for a sample drop on its way to asteroid B. Obviously, $\boldsymbol{q}_M$ is not the only mission plan; various other missions can be encoded in a similar manner.

It is clear that as the database of asteroids is increased from 2 to even just 10 or so, the digraph will quickly get unwieldy. In contrast, if we can generate a "physics" for the categorical variables, much the same way as differential equations are used to model the physics of continuous-time motion, then a routine generation of a physically

Received 13 February 2004; revision received 18 October 2004; accepted for publication 18 October 2004.

*Associate Professor, Code ME/Ro, Department of Mechanical and Astronautical Engineering; imross@nps.navy.mil. Associate Fellow AIAA.

†Senior Member Technical Staff, Aerospace Systems; dsouza@jsc.draper.com.

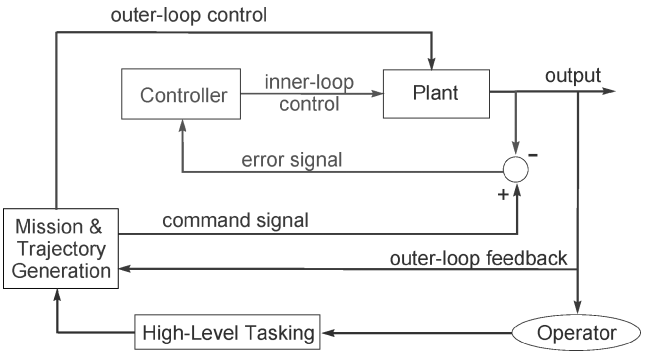

Fig. 1 Typical autonomous system architecture with operator in the outermost loop.

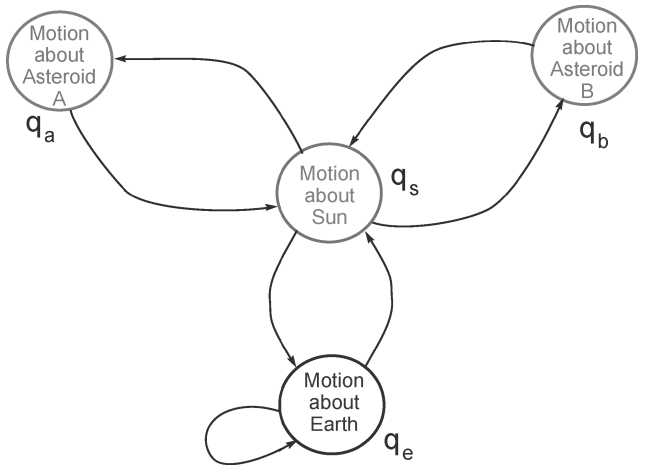

Fig. 2 Digraph for the asteroid mission planning problem for a database of two asteroids labeled A and B.

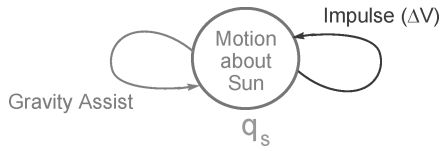

Fig. 3 Mission planning digraph for impulsive maneuvers.

realizable discrete event sequences should be possible, similar to the propagation of an ordinary differential equation (ODE). A procedure to model such a finite automaton in terms of the adjacency matrix of the associated digraph is described in Sec. II. In general, the finite automaton will be intricately interwoven with the ODE; hence, the finite state concepts cannot be handled independent of the ODE. Doing so will lead to trajectory segments that will be, at best, nonoptimal, and, at worst, infeasible when concatenated with other segments.[17] We defer a detailed discussion of this concept for the moment: It is evident that if we replace the word asteroid by the word planet in example 1, the ideas remain unchanged, except that the database of planets is only nine, whereas that for the near-Earth asteroids, for example, is several thousand. In this vein, mission planning to asteroids is far more complicated than mission planning to planets. Furthermore, in the case of mission planning for planetary encounters, the working database of planets can even be reduced from nine to just two or three because a mission to Mars will not involve a Mercury gravity assist or a Neptune flyby. As a result of a very small working database, planetary mission planning, particularly for impulsive maneuvers and instantaneous gravity assists, can be successfully performed in an ad hoc manner because the resulting digraph is quite simple, as shown in Fig. 3. In other words, the insightful mission planning tools of the past cannot be scaled up and automated to mission planning tools for the future. Thus, a new, formal framework is necessary to manage the complexities of

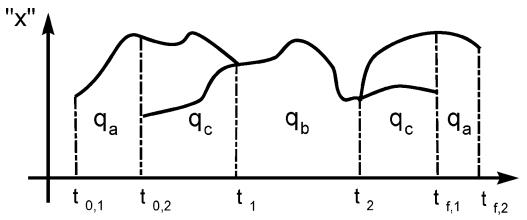

Fig. 4 Flight plan as a hybrid trajectory; ordinate labeled x is a generic continuous-time state variable.

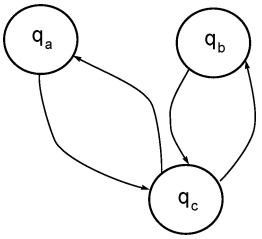

Fig. 5 Digraph for the refueling mission; there is no edge between the vertices q_a and q_b.

future missions. In this paper, we propose that the new language of hybrid optimal control theory provides this proper framework. To drive home this point, we consider a completely different mission planning problem next.

B. Example 2, Refuelling Mission

Two autonomous vehicles take off from possibly different sites at possibly different times, $(t_{0,1}, t_{0,2})$, then dock at some possibly unknown time, t_1, for some time interval, t_2–t_1, that may be fixed or free (for example, for refueling one of vehicles), and thereafter continue on toward their final destination that may end at different times, $(t_{f,1}, t_{f,2})$ (Fig. 4).

In this example, the two-vehicle system can be described as having three discrete states: one corresponding to a one-vehicle flight denoted q_a, another corresponding to a docked-vehicle flight denoted q_b, and the third corresponding to separated-vehicles flight denoted q_c. The categorical state space for this system is $\mathcal{Q} = \{q_a, q_b, q_c\}$. The digraph representing the discrete physics of the problem is shown in Fig. 5. A key difference between this digraph and the one corresponding to the asteroid problem is that, in Fig. 5, the vertices represent continuous-time dynamics of different dimensions. The vertex q_c represents continuous-time dynamics (ODEs) of twice the dimension, corresponding to two vehicles or agents, than that of the vertex q_a, which corresponds to a single-agent flight. Nonetheless, as in example 1, a mission plan can be modeled as a walk from one vertex to another. If $t_{0,1} = t_{0,2} \equiv t_0$ and $t_{f,1} = t_{f,2} \equiv t_f$, the flight plan calls for the execution

$$\boldsymbol{q}_1 = (q_c, q_b, q_c)$$

If $t_{0,1} \neq t_{0,2}$, then we have $\boldsymbol{q}_2 = (q_a, q_c, q_b, q_c)$. In addition, if $t_{f,1} \neq t_{f,2}$, then $\boldsymbol{q}_3 = (q_a, q_c, q_b, q_c, q_a)$. That there is no edge between the vertices q_a and q_b shows that a single-agent system cannot switch to a two-agent system without the presence of another agent (Fig. 4). As in the asteroid mission planning problem, the digraph shown in Fig. 5 will quickly get unwieldy as the number of agents increases from two to several hundred or more. Note that the cardinality of the categorical state space in this example ($N_Q = 3$) is not the same as the number of agents (two). In any case, all of the complexity issues discussed in the context of example 1 will apply to the large-scale multiagent problem as well. The prime customer for such large-scale multiagent intelligent systems is the military because it sees such systems as force multipliers, that is, a mechanism or process by which a military force, with fixed human capital, can achieve greater strength.

These examples are just a few of the vast array of even more complex mission designs being contemplated by civilian and military mission architects. For example, the asteroid mission planning problem becomes even more complex when the possibility of multiple

spacecraft is added to the mix, that is, a multiagent sample-return mission. Regardless, these examples illustrate why and how complex mission planning problems can be posed mathematically. Once this modeling task is performed, and if it is possible to solve these problems, it is clear that a systematic analysis of mission plans can be carried out and perhaps even automated toward the goal of creating intelligent systems. In this paper, we lay down the foundations for solving such problems and show, from the ground up, how the combination of some recently developed ideas from several disciplines hold the potential to automate sophisticated mission plans. The ideas that we combine in this paper are the emerging concepts in hybrid optimal control,[18–20] pseudospectral knotting methods,[5,21,22] and mixed-variable programming.[9,10] Because each of these topics is itself multidisciplinary, we emphasize that this paper does not purport to have conquered all of the issues involved in complex mission planning. Rather, we show, by way of a two-agent launch problem,[23] how the combination of certain emerging tools provides a promising approach to automate and optimize mission planning.

II. Hybrid Optimal Control Problem Formulation

Hybrid optimal control problems[18–20] are generalizations of ordinary optimal control problems in the sense that hybrid problems include categorical variables, as described in Sec. I. Categorical variables are discrete-valued variables, whereas variables that take values over a continuous set (such as $\mathbb{R}$), are continuous-valued variables. Note that discrete-valued variables do not necessarily take integer values; rather, they may take values from a list of categories[9] as described in the Appendix. Thus, whereas a continuous-valued variable may generate a continuous or discontinuous function of time, a discrete-valued variable generates a sequence that may be viewed as a piecewise-constant categorical function of time. Hybrid optimal control theory combines continuous systems with discrete systems, thus extending ordinary optimal control theory by including concepts from discrete event system theory.

Hybrid optimal control problems are not altogether new. In aerospace engineering, a special class of hybrid problems called multiphase optimal control problems[2] and branched optimization problems[17] has been pursued quite vigorously since the 1960s. Hybrid system theory generalizes and formalizes such problems by incorporating graph–theoretic concepts as outlined in Sec. I. This approach allows one to include mission planning as part of the problem formulation formally so that one can contemplate an algorithmic approach to problem solving. If these algorithms can be executed in real time, it is evident that it is possible to design an intelligent system by eliminating or reducing the tasks of the operator in the outer loop shown in Fig. 1. In pursuit of this goal, there has been considerable research activity over the last decade[24,25]; nevertheless, hybrid control is still in its relative infancy: A maximum principle for hybrid optimal control was formulated by Sussmann[18–20] only a few years ago, and a vast number of problems in hybrid system dynamics remain open.[26]

There are various competing approaches to modeling hybrid control systems.[18,27,28] In this paper, we adopt a particular framework based on Sussmann's approach.[18–20] In contrast, as indicated by Buss et al.,[28] the approach adopted by von Stryk and Glocker[14,15] and Buss et al.[28,29] is closer to that of Branicky et al.[27] Because our focus is computation, our formalism differs from Sussmann's in the sense that we coordinatize his geometric (coordinate-free) objects. In addition, we further articulate a model for a transition function $\mathcal{Q} \times \{0, 1\} \to \mathcal{Q}$ proposed in Ref. 22. As a result of many such differences, we describe our formalism in detail while we connect our approach to the other models discussed in the literature.

In a general hybrid optimal control problem, $\mathcal{Q}$ is some finite set of cardinality, $N_Q \in \mathbb{N}$, that represents the categorical state space, and $\boldsymbol{q}$ is a finite or infinite sequence of vertices that represents a walk. Because a major focus of our effort is computation, we limit our discussions to a finite sequence of locations in finite time [the so-called non-Zeno behavior (see Ref. 30)].

A. Continuous-Time Dynamics

Associated with each discrete state $q \in \mathcal{Q}$ is a continuous-time controlled dynamic system,

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, q) \tag{1}$$

where $\boldsymbol{f}(\cdot, q): \mathbb{R}^{N_x^q} \times \mathbb{R}^{N_u^q} \to \mathbb{R}^{N_x^q}$ is a Lipschitz-continuous function indexed by $\mathcal{Q}$, $N(\cdot) \in \mathbb{N}$, whereas $\boldsymbol{x} \in \mathbb{R}^{N_x^q}$ and $\boldsymbol{u} \in \mathbb{R}^{N_u^q}$ are the continuous-valued state and control variables, respectively. All of the discussions to follow also apply without change to nonautonomous dynamic systems of the form $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t, q)$, but, for notational simplicity, we prefer the autonomous dynamics given by Eq. (1). Also, note that for notational simplicity, we suppress the dependence of $\boldsymbol{x}$ and $\boldsymbol{u}$ on q.

B. Continuous-Valued State and Control Spaces

Similar to the vector field in Eq. (1), we define a hybrid, mixed, state-control constraint as

$$\boldsymbol{h}^L \le \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}, q) \le \boldsymbol{h}^U \tag{2}$$

where $\boldsymbol{h}(\cdot, q): \mathbb{R}^{N_x^q} \times \mathbb{R}^{N_u^q} \to \mathbb{R}^{N_h^q}$ is a Lipschitz-continuous function indexed by $\mathcal{Q}$, whereas $\boldsymbol{h}^L \in \mathbb{R}^{N_h^q}$ and $\boldsymbol{h}^U \in \mathbb{R}^{N_h^q}$ denote the lower and upper bounds on the values of the function $\boldsymbol{h}$. Equation (2) subsumes pure state constraints, pure control constraints, and the genuinely mixed state-control constraints. Thus, the pure-state and pure-control constraints incorporated in Eq. (2) may be written as, respectively,

$$\boldsymbol{h}_1^L \le \boldsymbol{h}_1(\boldsymbol{x}, q) \le \boldsymbol{h}_1^U \tag{3}$$

$$\boldsymbol{h}_2^L \le \boldsymbol{h}_2(\boldsymbol{u}, q) \le \boldsymbol{h}_2^U \tag{4}$$

where the subscripts on $\boldsymbol{h}$ imply vectors and functions of appropriate dimensions and dependence. Not only are Eqs. (3) and (4) subsumed under Eq. (2), but many of the attributes of the various models[27,28] for hybrid control can easily be mapped to our simpler descriptions. Thus, the constrained state space

$$\mathbb{X}(q) := \left\{\boldsymbol{x} \in \mathbb{R}^{N_x^q} : \boldsymbol{h}_1^L \le \boldsymbol{h}_1(\boldsymbol{x}, q) \le \boldsymbol{h}_1^U\right\} \tag{5}$$

is the same as the so-called invariant set used in electrical engineering,[26,27,30] and is often represented as the map

$$\mathbb{X} : \mathcal{Q} \to \mathcal{P}\left(\mathbb{R}^{N_x^q}\right) \tag{6}$$

where the notation $\mathcal{P}(\mathcal{A})$ denotes the power set[31] of $\mathcal{A}$, that is, $\mathcal{P}(\mathcal{A}) \equiv 2^{\mathcal{A}}$. We prefer our simpler representation of Eq. (5). In the same spirit, the control space is simply given by

$$\mathbb{U}(q) = \left\{\boldsymbol{u} \in \mathbb{R}^{N_u^q} : \boldsymbol{h}_2^L \le \boldsymbol{h}_2(\boldsymbol{u}, q) \le \boldsymbol{h}_2^U\right\} \tag{7}$$

Obviously, $\mathbb{U}: \mathcal{Q} \to \mathcal{P}(\mathbb{R}^{N_u^q})$. By letting the state and control spaces be jointly restricted, $\mathbb{X} \times \mathbb{U} \subseteq \mathbb{H}$, we can write

$$\mathbb{H}(q) = \left\{(\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^{N_x^q} \times \mathbb{R}^{N_u^q} : \boldsymbol{h}^L \le \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}, q) \le \boldsymbol{h}^U\right\} \tag{8}$$

Thus, $\mathbb{H}: \mathcal{Q} \to \mathcal{P}(\mathbb{R}^{N_x^q} \times \mathbb{R}^{N_u^q})$. It is clear that Eq. (2) and, hence, Eq. (8), are simpler in spirit, but somewhat more general than Eqs. (3–7). Note that mixed state-control constraints are widely considered to be difficult problems for both theory[32] and computation.[1] See Hartl et al.[33] for a thorough discussion.

C. Discrete Events

Arguably, the most important notion in hybrid control is the formalization and generalization of a switch. As noted earlier, we largely adopt Sussmann's generalizations because many of his ideas are rooted in his coordinate-free principles that are intrinsically general. From a practical point of view, it is necessary to "coordinatize" his geometric objects. Let $(\boldsymbol{x}, \boldsymbol{u})$ and $(\boldsymbol{x}', \boldsymbol{u}')$ denote the continuous-valued state and control variables associated with any

two vertices $q, q' \in \mathcal{Q}$. The generalized switching set, which may be empty, is called the event set $\mathbb{E}(q, q')$ and is defined (when nonempty) by means of an inequality constraint on a function $\boldsymbol{e}(\cdot, q, q')$: $\mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \times \mathbb{R} \times \mathbb{R}^{N_{x'}} \times \mathbb{R}^{N_{u'}} \times \mathbb{R} \to \mathbb{R}^{N_e^{qq'}}$, that is,

$$\mathbb{E}(q, q') = \{(\boldsymbol{x}, \boldsymbol{u}, \tau, \boldsymbol{x}', \boldsymbol{u}', \tau') : \boldsymbol{e}^L \leq \boldsymbol{e}(\boldsymbol{x}, \boldsymbol{u}, \tau, \boldsymbol{x}', \boldsymbol{u}', \tau', q, q') \leq \boldsymbol{e}^U\} \tag{9}$$

where $\boldsymbol{e}^L \in \mathbb{R}^{N_e^{qq'}}$ and $\boldsymbol{e}^U \in \mathbb{R}^{N_e^{qq'}}$ are the lower and upper bounds on the values of the function $\boldsymbol{e}(\cdot, q, q')$, respectively. The function $\boldsymbol{e}(\cdot, q, q')$ is called the event function[5,22] associated with the discrete states q and q'. In an event set, the clock is allowed to be reset, that is, it allows $\tau \neq \tau'$. This resetting of the clock allows us to treat the endpoint set and switching sets under a unified framework, in addition to simplifying the bookkeeping associated with a hybrid automaton.[19] In a computational method, the clock resets are used to exploit integrals of motion. Two recent examples of this can be found in the solar-sail mission design of Earth–Mars cyclers[34] and the double-rendezvous Mars sample-return problem.[35]

In investigating a transition of a hybrid system from one location, $q \in \mathcal{Q}$, to another location, $q' \in \mathcal{Q}$, at some (possibly unknown) time t_e, we define a switching set, $\mathcal{S}(q, q') \subset \mathbb{E}(q, q')$ as

$$\mathbb{E}(q, q') \supset \mathcal{S}(q, q') = \{(\boldsymbol{x}, \boldsymbol{u}, \tau, \boldsymbol{x}', \boldsymbol{u}', \tau') \in \mathbb{E}(q, q') : \tau = \tau' \equiv t_e \in \mathbb{R}\} \tag{10}$$

If $\mathcal{S}(q, q') \neq \emptyset$, then (q, q') is an edge of the digraph whose vertices are given by q and q'. Formally, the edges are defined as

$$\mathcal{E}_Q(q, q') = \{(q, q') \in \mathcal{Q} \times \mathcal{Q} : \mathcal{S}(q, q') \neq \emptyset\} \tag{11}$$

In defining all possible switching sets, the $N_Q \times N_Q$ adjacency matrix,[11] $\boldsymbol{A} = [A_{ij}]$,

$$A_{ij} = \begin{cases} 1 & \text{if} \quad \mathcal{S}(q_i, q_j) \neq \emptyset \\ 0 & \text{if} \quad \mathcal{S}(q_i, q_j) = \emptyset \end{cases} \tag{12}$$

turns out to be a very useful concept, as elaborated on in Sec. III and as described for the two-agent launch problem in Sec. IV. Essentially, we use $\boldsymbol{A}$ as a computational means to encode the digraph of the hybrid automaton.

If the states are continuous when the system transitions from one location q to another location q', the continuity set $\mathcal{C}(q, q')$ is defined as

$$\mathbb{E}(q, q') \supset \mathcal{C}(q, q') = \{(\boldsymbol{x}, \boldsymbol{u}, \tau, \boldsymbol{x}', \boldsymbol{u}', \tau') \in \mathbb{E}(q, q') : \boldsymbol{x} = \boldsymbol{x}', \tau = \tau'\} \tag{13}$$

This set obviously implies that $N_x^q = N_x^{q'}$. Because state continuity occurs frequently, the continuity set is exploited within the context of pseudospectral (PS) knots as a "soft knot."[5,21]

To clarify how the boundary conditions can be formally treated as an event condition, let $t_0 \in \mathbb{R}$ be the initial time and suppose that we have $N_s \in \mathbb{Z}_+$ finite switches. Let $N_{s1} := N_s + 1$, and let $t_{N_{s1}} \equiv t_f > t_0$ be the final time. Let $q^0 \in \mathcal{Q}_0 \subseteq \mathcal{Q}$ be the initial-time condition for the discrete state, and let $q^{N_s} \in \mathcal{Q}_f \subseteq \mathcal{Q}$ be the final-time condition. The boundary conditions for this problem are typically written as

$$\boldsymbol{e}_0^{L_0} \leq \boldsymbol{e}_0(\boldsymbol{x}_0, t_0) \leq \boldsymbol{e}_0^{U_0} \qquad q^0 \in \mathcal{Q}_0 \tag{14}$$

$$\boldsymbol{e}_f^{L_f} \leq \boldsymbol{e}_f(\boldsymbol{x}_f, t_f) \leq \boldsymbol{e}_f^{U_f} \qquad q^{N_s} \in \mathcal{Q}_f \tag{15}$$

As noted elsewhere,[5,34,35] in many problems the initial-time and final-time conditions are not decoupled as the preceding equations suggest. Rather, they are inextricably coupled, for example, launch windows for interplanetary travel. Hence, we prefer not to separate the boundary conditions. Separate or otherwise, the boundary conditions can be framed as part of the event set $\mathbb{E}(q, q')$ by taking $q = q^{N_s}$ and $q' = q^0$. That is, we can write

$$\left(\boldsymbol{x}_f, \boldsymbol{u}_f, t_f, \boldsymbol{x}_0, \boldsymbol{u}_0, t_0, q^{N_s}, q^0\right) \in \mathbb{E}(q^{N_s}, q^0) \tag{16}$$

Thus, the clock variables τ and τ' do not play the role of a switch time, but rather that of the final and initial time; hence, $\tau \neq \tau'$ [cf. Eq. (10)]. In most cases, the controls are not part of the boundary conditions; thus, the functional dependence on $\boldsymbol{u}$ in Eq. (16) can be safely ignored.

As noted in Ref. 22, the event set generalizes the notion of a guard, $\mathcal{G}_i$: $\mathcal{Q} \times \mathcal{Q} \to \mathcal{P}(\mathbb{R}^{N_x^q})$, and reset map, $\mathcal{R} : \mathcal{Q} \times \mathcal{Q} \times \mathbb{R}^{N_x^q} \to \mathcal{P}(\mathbb{R}^{N_x^q})$, which are commonly used concepts in electrical engineering[24–27,30]; these maps are given by

$$\mathcal{G}(q, q') = \left\{\boldsymbol{x} \in \mathbb{R}^{N_x^q} : (\boldsymbol{x}, \boldsymbol{u}, \tau, \boldsymbol{x}', \boldsymbol{u}', \tau') \in \mathcal{S}(q, q')\right\}$$

$$\mathcal{R}(q, q', \boldsymbol{x}) = \left\{\boldsymbol{x}' \in \mathbb{R}^{N_x^{q'}} : (\boldsymbol{x}, \boldsymbol{u}, \tau, \boldsymbol{x}', \boldsymbol{u}', \tau') \in \mathcal{S}(q, q')\right\}$$

Thus, in the asteroid mission design problem elaborated on in Sec. I, we may define guard sets in terms of spheres of influence, whereas the reset map, as already discussed in Sec. I, may correspond to mass drops, impulsive maneuvers, or instantaneous gravity assists.

D. Finite Automata

In much of the hybrid control literature,[24–27,30] the finite dynamics are assumed to be given in terms of a transition function $q \mapsto q'$. Because much of our focus is on computation, it is necessary to have a practical means of modeling a transition function that is in accordance with our hybrid model, as well as with our computational approach (described in Sec. III). Consistent with our approach of solving ordinary optimal control problems using global methods, we view a finite automaton in terms of an execution $\boldsymbol{q}$ as a fundamental object. To facilitate a generation of $\boldsymbol{q}$ in terms of matrix operations, we elaborate the $*$ operation defined in Ref. 22. This operation was inspired by the work of von Stryk and Glocker[14,15] and Buss et al.[28]

Let $[Q]$ to be a row matrix whose columns are the N_Q elements of $\mathcal{Q}$. To model a transition map $[Q] \mapsto \boldsymbol{q}$, we use the binary set $\{0, 1\}$ to define an operation $*$ over the Cartesian product $\mathcal{Q} \times \{0, 1\}$ as[22]

$$q * 0 = \emptyset, \qquad q * 1 = q \qquad \forall q \in \mathcal{Q}$$

Let $\mathbb{D}^{n \times m} \subset \{0, 1\}^{n \times m}$, $n, m \in \mathbb{N}$, be the set of $n \times m$ matrices defined as

$$\mathbb{D}^{n \times m} := \left\{\Delta \in \{0, 1\}^{n \times m} : \sum_{i=1}^{n} \Delta_{ij} \in \{0, 1\} \quad \forall j = 1, \ldots, m\right\} \tag{17}$$

This column-sum property allows us to define $\Delta \in \mathbb{D}^{N_Q \times N_{s1}}$ as our discrete control matrix, provided that we now define[22]

$$q + \emptyset = q = \emptyset + q, \qquad \emptyset + \emptyset = \emptyset$$

With these intuitive notions, the $*$ operation is now extended to the product $[Q] * \Delta$ in the usual sense of a matrix operation with $\Delta \in \mathbb{D}^{N_Q \times N_{s1}}$. Thus, $[Q] * \Delta$ generates a sequence of N_{s1} elements; hence, we can write

$$\boldsymbol{q} = [Q] * \Delta, \qquad \Delta \in \mathbb{D}^{N_Q \times N_{s1}} \tag{18}$$

as the fundamental equation to generate $\boldsymbol{q}$.

The set $\mathbb{D}$ plays two key roles in generating the finite sequence. Because the operation $q + q'$ is not defined, and may, in fact, be meaningless (because q and q' are categorical variables), it is naturally excluded when Δ is restricted to lie in the set $\mathbb{D}^{N_Q \times N_{s1}}$. By allowing the column sum of Δ to be zero, we facilitate a computationally "elastic" approach to generating the sequence $\boldsymbol{q}$. That is, if the jth column of Δ sums up to zero, then the subsequence, $[q^{j-1}, \emptyset, q^{j+1}]$, is understood to be equal to $[q^{j-1}, q^{j+1}]$.

For Eq. (18) to generate feasible sequences for a hybrid automaton, the discrete controller Δ must be further restricted to

a set $\mathbb{U}_D$, much in the same way as the continuous-time control space $\mathbb{U}(q)$ was restricted earlier [cf. Eq. (7)]. Deferring the details of the computational aspects of $\mathbb{U}_D$ to Sec. III, it is apparent that we can generate an execution by way of the finite dynamics,

$$\boldsymbol{q} = [Q] * \Delta, \qquad \Delta \in \mathbb{U}_D \subseteq \mathbb{D}^{N_Q \times N_{s1}} \tag{19}$$

E. Cost Functions

Associated with any pair, $(q, q') \in \mathcal{Q} \times \mathcal{Q}$, is an event cost defined by the function

$$E(\cdot, q, q') : \mathbb{E}(q, q') \to \mathbb{R} \cup \{\infty\} \tag{20}$$

The cost function $E(\cdot, q, q')$ takes the value infinity whenever $\mathbb{E}(q, q') = \emptyset$. In the computational scheme described in Sec. III, we handle the evaluation of infinity for the switching set $\mathcal{S}(q, q')$ by way of the adjacency matrix; that is, whenever $\mathcal{S}(q, q') = \emptyset$, a transition from q to q' is blocked. Thus, Eq. (20) generalizes the Mayer cost associated in ordinary optimal control theory and can be viewed as a weight of the digraph. Similarly, in generalizing a Lagrange cost, we associate a running cost, that is, the integrand of a Lagrange cost functional, to each location $q \in \mathcal{Q}$,

$$F(\cdot, q) : \mathbb{R}^{N_x^q} \times \mathbb{R}^{N_u^q} \to \mathbb{R} \tag{21}$$

To formulate a hybrid Bolza cost functional, we let $\boldsymbol{q} = [q^0, q^1, \ldots, q^{N_s}]$ be a finite sequence of locations where $q^j \in \mathcal{Q}$ for $j = 0, 1, \ldots, N_s$. In an ordinary optimal control problem, N_s would be zero. Let $\boldsymbol{a} = [a_0, a_1, \ldots, a_{N_s}]$ and $\boldsymbol{b} = [b_0, b_1, \ldots, b_{N_s}]$ be real-valued matrices representing finite sequences of real numbers associated with $\boldsymbol{q}$ such that $[a_i, b_i]$, $a_i \neq b_i$, are nonzero intervals in $\mathbb{R}$. We define the initial time as $t_0 = a_0$ and the final time as $t_f = b_{N_s}$. Usually, we will have $a_{i+1} = b_i$ (as in the case of a switch), but, as noted earlier, we reserve the freedom to not make this assumption as a result of the theoretical and computational benefits it offers. Let $\boldsymbol{x}(\cdot) : t \mapsto (\boldsymbol{x}^0, \boldsymbol{x}^1, \ldots, \boldsymbol{x}^{N_s})$ and $\boldsymbol{u}(\cdot) : t \mapsto (\boldsymbol{u}^0, \boldsymbol{u}^1, \ldots, \boldsymbol{u}^{N_s})$ represent the continuous-time state and control functions associated with $\boldsymbol{q}$, so that for each q^j we have $\boldsymbol{x}^j \in \mathbb{R}^{N_x^{q^j}}$ and $\boldsymbol{u}^j \in \mathbb{R}^{N_u^{q^j}}$. As in standard optimal control theory,[32] the function space $\mathcal{X}$ from which we select the functions, $t \mapsto \boldsymbol{x}^j$, is the space of absolutely continuous functions, whereas the space $\mathcal{U}$ from which we select $t \mapsto \boldsymbol{u}^j$ is the space of measurable functions. If the optimal solutions are smoother than these function spaces, as is expected to be the case for physical problems (assuming models of sufficient fidelity), then our computational method naturally exploits this property. This is further elaborated on in Sec. III. The tuple, $[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{q}, \Delta, N_s]$, is the hybrid system trajectory. To assist in an efficient problem formulation, we follow Sussmann[20] and define $\tilde{+}$ as

$$j \,\tilde{+}\, 1 = \begin{cases} j+1 & \text{if} \quad j < N_s \\ 0 & \text{if} \quad j = N_s \end{cases} \tag{22}$$

This operation simply allows us to wrap indices because q^j, $j = 0, 1, \ldots, N_s$, is equal to $q^{j \,\tilde{+}\, 1}$, $j = N_s, 0, 1, \ldots, N_s - 1$. For any given hybrid system trajectory, the Bolza cost functional, $J : (\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{q}, \Delta, N_s) \mapsto \mathbb{R}$, can be evaluated from

$$\begin{aligned} &J[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{q}, \Delta, N_s] \\ &\quad = \sum_{j=0}^{N_s} \Bigg(E\big(\boldsymbol{x}^j(b_j), \boldsymbol{u}^j(b_j), b_j, \boldsymbol{x}^{j \,\tilde{+}\, 1}(a_{j \,\tilde{+}\, 1}), \boldsymbol{u}^{j \,\tilde{+}\, 1}(a_{j \,\tilde{+}\, 1}), \\ &\qquad a_{j \,\tilde{+}\, 1}, q^j, q^{j \,\tilde{+}\, 1}\big) + \int_{a_j}^{b_j} F(\boldsymbol{x}(t), \boldsymbol{u}(t), q^j)\, \mathrm{d}t \Bigg) \end{aligned} \tag{23}$$

F. Problem Formulation

The hybrid Bolza optimal control problem is succinctly formulated as

$$(\mathcal{H}) \begin{cases} \text{Minimize} & J[\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{q}, \Delta, N_s] \\ \text{Subject to} & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), q^j) \\ & \boldsymbol{q} = [Q] * \Delta \\ & (\boldsymbol{x}(t), \boldsymbol{u}(t)) \in \mathbb{H}(q^j) \\ & \Delta \in \mathbb{U}_D \\ & \big(\boldsymbol{x}^j(b_j), \boldsymbol{u}^j(b_j), b_j, \\ & \boldsymbol{x}^{j \,\tilde{+}\, 1}(a_{j \,\tilde{+}\, 1}), \boldsymbol{u}^{j \,\tilde{+}\, 1}(a_{j \,\tilde{+}\, 1}), a_{j \,\tilde{+}\, 1}, \\ & q^j, q^{j \,\tilde{+}\, 1}\big) \in \mathbb{E}(q^j, q^{j \,\tilde{+}\, 1}) \\ & \text{a.e. } t, q^j \in \mathcal{Q}, \qquad j = 0, 1, \ldots, N_s \end{cases}$$

It is apparent that problem $\mathcal{H}$ subsumes the notion of a multiphase optimal control problem.[2] Obviously, if $N_s = 0$, we recover an ordinary Bolza problem. In addition to the real variables, problem $\mathcal{H}$ contains the categorical variable sequence $\boldsymbol{q}$, a binary decision variable matrix Δ, and a nonnegative integer N_s.

III. Solution Methods

Many issues regarding methods for solving hybrid optimal control problems parallel that of ordinary optimal control problems. A practical method for solving nonlinear hybrid optimal control problems is one of the great challenges in computation. In a recent survey, Xu and Antsaklis[36] note the limitations of many proposed methods. Von Stryk and Glocker,[15] Glocker and von Stryk,[16] and Buss et al.[28] suggest a decomposition method using branch and bound. In a brief complement to these perspectives, we note that if a particular method has difficulties in solving ordinary optimal control problems, it will undoubtedly have even more severe problems in solving hybrid problems. In this spirit, despite advancements in viscosity methods,[37] we rule out the Hamilton–Jacobi framework because it is beset with fundamental problems of nonsmoothness of the value function, as well as Bellman's famous "curse of dimensionality." A computational method based on the hybrid minimum principle will inherit all of the problems of indirect methods.[1,2] As is now well known,[1–5] direct methods and, in particular, direct collocation methods, offer the best chance of solving complex optimal control problems. In following this perspective, we elaborate on a computational method to model the discrete control space $\mathbb{U}_D$.

To generate an execution $\boldsymbol{q}$ from the finite dynamics, $\boldsymbol{q} = [Q] * \Delta$, the first column of Δ must be consistent with $\mathcal{Q}_0 \subseteq \mathcal{Q}$, the specified set of initial discrete states. Because $\mathcal{Q}$ is finite, it is equivalent to a finite subset of $\mathbb{N}$ (in the sense of set equivalence[31]); hence, we can define an N_Q column vector $\boldsymbol{A}^0$ with elements in $\{0, 1\}$ that represents $\mathcal{Q}_0$, with 1 indicating membership in $\mathcal{Q}_0$ and 0 otherwise. For example, if $\mathcal{Q}_0 = \mathcal{Q}$, then, $\boldsymbol{A}^0$ is the vector of ones. Thus, the selection of the first column of $\Delta \in \mathbb{D}^{N_Q \times N_{s1}}$ can be decided by

$$\Delta_{i,1} \in \big\{A_i^0, 0\big\}, \qquad i = 1, \ldots, N_Q \tag{24}$$

In the same way, we can establish a bijection between $\mathcal{Q}_f \subseteq \mathcal{Q}$ and the last column of Δ by defining an N_Q column vector $\boldsymbol{A}^f$; that is,

$$\Delta_{i,N_{s1}} \in \big\{A_i^f,\ 0\big\}, \qquad i = 1, \ldots, N_Q \tag{25}$$

The decision on filling the interior columns of Δ must be consistent with (q^j, q^{j+1}), $j = 1, \ldots, N_s$, being an edge of the digraph associated with the hybrid automaton, that is, we must have $\mathcal{S}(q^j, q^{j+1}) \neq \emptyset$. Because this information is encoded in the adjacency matrix [cf. Eq. (12)], we can write

$$\begin{aligned} &\Delta_{i,j+1} \in \{A_{ki},\ 0\} \qquad \text{for} \qquad \Delta_{k,j} = 1 \\ &\qquad i = 1, \ldots, N_Q, \qquad j = 1, \ldots, N_s \end{aligned} \tag{26}$$

Thus, the set of allowable discrete controls $\mathbb{U}_D$ can be defined as

$$\mathbb{U}_D := \left\{ \Delta \in \mathbb{D}^{N_Q \times N_{s1}} : \Delta_{i,j}, \text{satisfy Eqs. (24–26)} \right.$$
$$\left. \text{for} \quad i = 1, \ldots, N_Q, \quad j = 1, \ldots, N_{s1} \right\} \qquad (27)$$

To generate an execution, we have to solve the following feasible integer programming (FIP) problem,

$$\text{(FIP)} \begin{cases} \text{Find} & \Delta,\ N_{s1} \\ \text{Subject to} & \Delta \in \mathbb{U}_D \subseteq \mathbb{D}^{N_Q \times N_{s1}} \\ & N_{s1} \leq N_{s,\max} \in \mathbb{N} \end{cases}$$

where $N_{s,\max}$ is some finite upper bound that reflects the maximum number of allowed switches. Thus, Zeno problems are disallowed as with any computational method. Note that problem FIP is significantly simpler than a standard integer programming problem. The feasible space for problem FIP grows only as $(N_Q)^{N_{s1}}$. In the absence of the set $\mathbb{D}$, the complexity of the problem grows as $2^{N_Q \times N_{s1}}$. A similar observation has been made by von Stryk (see Refs. 14 and 15) regarding their motorized traveling salesman problem. They also indicate the NP completeness of such problems.

A solution to problem FIP implies a feasible triple $(\boldsymbol{q}, \Delta, N_s)$. Every such feasible triple reduces problem $\mathcal{H}$ to a problem over real variables. Thus, a prerequisite for solving a hybrid optimal control problem is an efficient method to solve a Bolza problem subject to event constraints. Although it is not the only viable method, we use the PS knotting method[5,21] because of its ready availability within the MATLAB® problem solving environment by way of the software package DIDO.[38] PS knots provide an efficient approach to incorporate discrete events in a Bolza problem. Details of this method are described in Refs. 5, 21, and 22. Here, we clarify this approach by explaining the concept of PS knots from the point of view of representing a hybrid automaton.

The PS knotting method essentially consists of two major steps. In the first step, the problem is discretized at each location, $q \in \mathcal{Q}$, by a standard PS method [like the Legendre–PS method (see Refs. 39 and 40)] wherein the cost function, dynamics, and path constraints are all discretized over an appropriately chosen set of nodes (such as the Legendre–Gauss–Lobatto nodes). The basic idea behind PS methods is the use of Lagrange interpolating functions for discretizing the continuous-time states and control variables and the use of optimal quadrature points such as the extrema of orthogonal polynomials. Typical node selections are based on Legendre–Gauss–Lobatto or Chebyshev–Gauss–Lobatto points. The state equations, path constraints, and the boundary conditions are imposed at these nodes. This approach offers significant advantages over other spectral methods [Galerkin or tau (see Ref. 41)] in the ease of handling nonlinearities, coupled boundary conditions, and mixed constraints that arise in practical optimal control problems. There is no need for lengthy calculations of matrices based on weak variation of the problem that is required in the Galerkin or tau methods. Any spectral method offers exponential accuracy[41] in the approximation of underlying functions (when they are sufficiently smooth) as opposed to the finite order accuracy of finite difference or finite element methods. If the state and control functions are sufficiently smooth (over each location q) then the spectral accuracy of the discretization method allows one to choose a substantially lower number of nodes than other methods, such as Runge–Kutta, to generate a specified degree of approximation (see Ref. 42). See Ref. 43 for a recent numerical comparison pertaining to ascent guidance.

In the second step, the event conditions are imposed over the Lobatto nodes. Because the Lobatto nodes can be collocated over a single point (Fig. 6), they facilitate an accurate representation of the finite automaton in terms of imposing the event conditions over the double Lobatto points. To distinguish such nodes from the interior nodes, these points are also known as PS knots.[5] Thus, PS knots transfer information across the nodes by using the event conditions (called knotting conditions[5] within the context of the discretized problem), thereby completing the full discretization of the hybrid

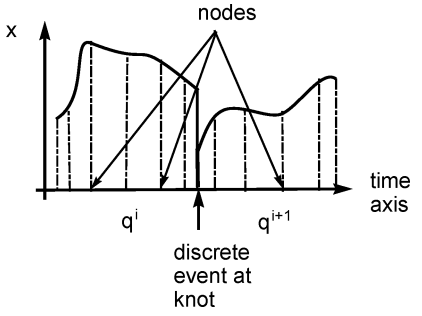

Fig. 6 Discretization of a hybrid system trajectory; knots are double Lobatto node points where a discrete event occurs.

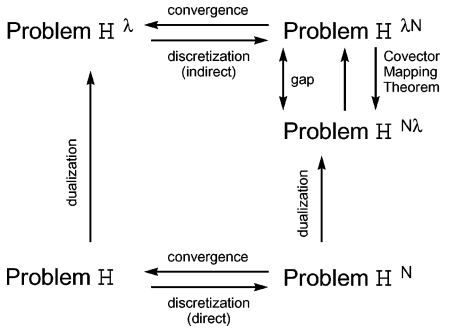

Fig. 7 Solving hybrid optimal control problems (adapted from Refs. 3, 22, and 40).

problem. Thus, the discretized problem is a mixed-variable programming (MVP) problem. In general, an MVP cannot be solved by a mixed integer programming approach because relaxations for branch and bound are not available.[9] Further, because $\mathcal{Q}$ is not a metric space, fundamental notions, such as the neighborhood of a point, are not defined. Although it is possible to define a discrete metric,[31] $d(\cdot, \cdot)$ over $\mathcal{Q}$, as

$$d(q, q') = \begin{cases} 0 & \text{if} \quad q = q' \\ 1 & \text{if} \quad q \neq q' \end{cases}$$

the problems are far from completed, and they are beyond the scope of this paper. See Ref. 9 for a recent discussion. Regardless of these issues, we note that the MVP, labeled problem $\mathcal{H}^N$ in Fig. 7, is unquestionably easier to solve than the mixed variable, mixed complementary problem (problem $\mathcal{H}^{\lambda N}$) that results from a discretization of problem $\mathcal{H}^{\lambda}$, where problem $\mathcal{H}^{\lambda}$ is a generalized root-finding problem that results from an application of the hybrid minimum principle. A thorough discussion of these concepts can be found in Refs. 22 and 40, and a broader description of the ideas is presented in Ref. 3. For the purposes of brevity, we do not state the hybrid necessary conditions here, but rather note that a simplified form of Sussmann's version can be stated in a familiar Pontryagin style (see Ref. 22). The gap shown in Fig. 7 is a result of dualization and discretization being typically noncommutative.[3,40] In the case of the PS knotting method, the gap is closed by a change in coordinates facilitated by the hybrid closure conditions,[22] which are a generalization of the conditions described in Ref. 40. In essence, the easiest approach to solve a general nonlinear hybrid optimal control problem is to solve the MVP in a manner that permits dualization to commute with discretization.[3,22,40] Despite that methods for solving MVPs are at an incipient stage,[9,10] the need to solve hybrid problems efficiently provides new motivation for advancements in large-scale, mixed-variable optimization.

In the next section, we illustrate the details of our ideas by way of a two-agent launch problem.[23]

IV. Illustrative Problem

The motivation and many details of a two-agent launch problem are discussed in Refs. 23 and 44; hence, only those details that are

pertinent to demonstrating the main features of the hybrid control framework are described in this section. We briefly note that not all multiagent systems have a natural hybrid structure; for example, the spacecraft formation design and control problems posed in Refs. 45 and 46 are not hybrid problems. A launch problem similar to the one described in this section has been solved by Weigel and Well[47] for a specified mission plan using a direct shooting method. A single-agent version of this problem using PS methods is discussed in Ref. 48 and by Rea.[49] Rea[49] also discusses guidance via real-time optimization. Other examples of hybrid optimal control problems arising in space mission design are the solar-sail problems discussed in Refs. 34 and 35, although the hybrid structures of these problems are fairly simple. Among the hybrid control problems typically found in the literature,[24–26] the problem discussed in this section appears to be one of the most complex in the following respects: the high dimension of the state variables (14), dimensional switching of the state variables (7–14), the high dimension of the control variables (6), dimensional switching of the control variables (3–6), the cardinality of the discrete state space (7), nonlinearities in the continuous-time dynamics, nonlinear control constraints, and nonlinear state constraints. The full practical problem has even greater complexity, as described in Ref. 44.

A. Preliminary Problem Statement

Two vehicles are mated together, similar to a staged launch vehicle, except that the stages are also controlled vehicles as opposed to just dropped masses. Solid propellants are used for the propulsion system. The problem is to transfer the ascent vehicle to orbit while transporting the return vehicle to a drop-down region (a return point constraint). The overall problem is to explore various concepts of operations (CONOPS) to support the design optimization of the system.

B. Discrete States

It can be shown[44] that the cardinality of the discrete state space for this problem is 12. For the purpose of brevity, we consider a seven-element state space as outlined in the Appendix.

We define a two-dimensional categorical variable q where one variable represents the information on the state of the mating, whereas the other variable represents the state of thrusting. As developed in Ref. 44 and summarized in the Appendix, the categorical state space $\mathcal{Q}$ is defined as the seven-element set,

$$\mathcal{Q} = \left\{ \begin{pmatrix} \text{mated} \\ \text{off} \end{pmatrix}, \begin{pmatrix} \text{mated} \\ \text{on} \end{pmatrix}, \begin{pmatrix} \text{separated} \\ \text{off} \end{pmatrix}, \begin{pmatrix} \text{separated} \\ \text{on/off} \end{pmatrix}, \begin{pmatrix} \text{mated} \\ \text{on/off} \end{pmatrix}, \begin{pmatrix} \text{single} \\ \text{on} \end{pmatrix}, \begin{pmatrix} \text{single} \\ \text{off} \end{pmatrix} \right\} \tag{28}$$

where (mated, off) is the state corresponding to the two vehicles flying (including no-flight or launch pad case) as a single agent, that is, mated, at zero thrust; (separated, off) corresponds to the two-vehicle state at zero thrust; (mated, on) corresponds to the mated vehicle at full thrust; (mated, on/off) corresponds to the mated vehicle with one engine on and the other engine off; (separated, on/off) is the state corresponding to separated vehicles with one vehicle at nonzero thrust and the other at zero; (single, on) corresponds to the case when only one vehicle is pertinent, and it has nonzero thrust; and (single, off) represents the situation when the pertinent vehicle is at zero thrust. Note that, true to the behavior of solid propellants, thrust is some function of the categorical state variable and not artificially relaxed to be part of the continuous vector control. Note also that although the cardinality of $\mathcal{Q}$ is seven and, thus, $\mathcal{Q} \sim \{1, 2, 3, 4, 5, 6, 7\}$, no evaluations are possible for relaxations of the set, $\{1, \ldots, 7\}$ (as commonly done in integer programming) because $\mathcal{Q}$ is a space of linguistic information. In an object-oriented problem-solving environment such as MATLAB, such information can be easily coded by using characters instead of numbers for the variables, and $\mathcal{Q}$ can be represented in terms of cell arrays. Example applications of such high-level formulations are given in Refs. 12, 38, and 44. For notational simplicity, we adopt the following symbols:

$$q_a = \begin{pmatrix} \text{mated} \\ \text{off} \end{pmatrix}, \quad q_b = \begin{pmatrix} \text{mated} \\ \text{on} \end{pmatrix}, \quad q_c = \begin{pmatrix} \text{separated} \\ \text{off} \end{pmatrix}$$

$$q_d = \begin{pmatrix} \text{separated} \\ \text{on/off} \end{pmatrix}, \quad q_e = \begin{pmatrix} \text{single} \\ \text{on} \end{pmatrix}$$

$$q_f = \begin{pmatrix} \text{single} \\ \text{off} \end{pmatrix}, \quad q_g = \begin{pmatrix} \text{mated} \\ \text{on/off} \end{pmatrix} \tag{29}$$

C. Continuous-Time Dynamics

For $q \in \{q_a, q_b, q_e, q_f, q_g\}$, we have the following dynamics with $\boldsymbol{x} = [\boldsymbol{r}, \boldsymbol{v}, m] \in \mathbb{R}^7$ and $\boldsymbol{u} \in \mathbb{R}^3$,

$$\dot{\boldsymbol{r}} = \boldsymbol{v} \tag{30}$$

$$\dot{\boldsymbol{v}} = \frac{T(q)}{m}\boldsymbol{u} - \frac{\mu}{r^3}\boldsymbol{r} - \frac{\rho(r)C_D(q)S(q)v_{\text{rel}}(\boldsymbol{r}, \boldsymbol{v})\boldsymbol{v}_{\text{rel}}(\boldsymbol{r}, \boldsymbol{v})}{2} \tag{31}$$

$$\dot{m} = \gamma(q) \tag{32}$$

where $T(q)$ and $\gamma(q)$ are thrust and mass flow rates, respectively,

$$(T(q), \gamma(q)) = \begin{cases} (0, 0) & \text{if} \quad q \in \{q_a, q_f\} \\ (T_b, \gamma_b) & \text{if} \quad q = q_b \\ (T_e, \gamma_e) & \text{if} \quad q = q_e \\ (T_g, \gamma_g) & \text{if} \quad q = q_g \end{cases} \tag{33}$$

and (T_b, T_e, T_g) and $(\gamma_b, \gamma_e, \gamma_g)$ are some given positive values of thrust and mass flow rates, respectively. Similarly, $C_D(q)$ is the coefficient of drag,

$$C_D(q) = \begin{cases} C_{D,2} & \text{if} \quad q \in \{q_a, q_b, q_g\} \\ C_{D,1} & \text{if} \quad q \in \{q_e, q_f\} \end{cases} \tag{34}$$

where the numerical subscripts on C_D denote some constant positive values. Similarly, $S(q)$ is the reference area,

$$S(q) = \begin{cases} S_2 & \text{if} \quad q \in \{q_a, q_b, q_g\} \\ S_1 & \text{if} \quad q \in \{q_e, q_f\} \end{cases} \tag{35}$$

where $S_1 > 0$ and $S_2 > 0$ are some given numbers.

The components of the control $\boldsymbol{u}$ are the thrust attitude direction cosines. With regard to the other symbols in Eqs. (30–32), $\boldsymbol{r}$ is the position vector, $\boldsymbol{v}$ is the inertial velocity vector, m is the vehicle mass, μ is the gravitational constant, and $\boldsymbol{v}_{\text{rel}}(\boldsymbol{r}, \boldsymbol{v})$ is the velocity of the vehicle with respect to the atmosphere,

$$\boldsymbol{v}_{\text{rel}}(\boldsymbol{r}, \boldsymbol{v}) := \boldsymbol{v} - \boldsymbol{\Omega}_{\oplus} \times \boldsymbol{r} \tag{36}$$

where $\boldsymbol{\Omega}_{\oplus}$ is the rotational velocity of the atmosphere,

$$v_{\text{rel}}(\boldsymbol{r}, \boldsymbol{v}) := |\boldsymbol{v}_{\text{rel}}(\boldsymbol{r}, \boldsymbol{v})| \tag{37}$$

and $|\cdot|$ denotes the Euclidean norm. The atmospheric density $\rho(r)$ is modeled as

$$\rho(r) = \rho_0 \exp\left[(R_{\oplus} - r)/H_s\right] \tag{38}$$

where ρ_0 is the atmospheric density at sea level, $R_{\oplus}$ is the radius of the Earth, and H_s is the scale height parameter.

For $q \in \{q_c, q_d\}$, we use subscripts A and R to denote the ascent vehicle and the return vehicle, respectively. Hence, the state and control variables can be written as $\boldsymbol{x} = [\boldsymbol{r}_A, \boldsymbol{r}_R, \boldsymbol{v}_A, \boldsymbol{v}_R, m_A, m_R] \in \mathbb{R}^{14}$ and $\boldsymbol{u} = [\boldsymbol{u}_A, \boldsymbol{u}_R] \in \mathbb{R}^6$. Thus, we have the following dynamics:

$$\dot{\boldsymbol{r}}_A = \boldsymbol{v}_A \tag{39}$$

$$\dot{\boldsymbol{r}}_R = \boldsymbol{v}_R \tag{40}$$

$$\dot{\boldsymbol{v}}_A = \frac{T_A(q)}{m}\boldsymbol{u}_A - \frac{\mu}{r_A^3}\boldsymbol{r}_A - \frac{\rho(r_A)C_{D,1}S_1 v_{\text{rel}}(\boldsymbol{r}_A,\boldsymbol{v}_A)\boldsymbol{v}_{\text{rel}}(\boldsymbol{r}_A,\boldsymbol{v}_A)}{2} \quad (41)$$

$$\dot{\boldsymbol{v}}_R = \frac{T_R(q)}{m}\boldsymbol{u}_R - \frac{\mu}{r_R^3}\boldsymbol{r}_R - \frac{\rho(r_R)C_{D,1}S_1 v_{\text{rel}}(\boldsymbol{r}_R,\boldsymbol{v}_R)\boldsymbol{v}_{\text{rel}}(\boldsymbol{r}_R,\boldsymbol{v}_R)}{2} \quad (42)$$

$$\dot{m}_A = \gamma_A(q) \quad (43)$$

$$\dot{m}_R = \gamma_R(q) \quad (44)$$

where

$$(T_A(q), T_R(q)) = \begin{cases} (0,0) & \text{if} \quad q = q_c \\ (T_d, 0) & \text{if} \quad q = q_d \end{cases} \quad (45)$$

$$(\gamma_A(q), \gamma_R(q)) = \begin{cases} (0,0) & \text{if} \quad q = q_c \\ (\gamma_d, 0) & \text{if} \quad q = q_d \end{cases} \quad (46)$$

and where $T_d > 0$ and $\gamma_d > 0$ are some given values of thrust and mass flow rates, respectively.

D. Continuous-Valued State and Control Spaces

As discussed elsewhere,[47–49] the continuous state spaces are constrained due to the various structural and thermal constraints on the vehicles. For the purposes of brevity, we consider only the dynamic pressure constraints of the form, $0 \le h(\boldsymbol{r},\boldsymbol{v}) \le h^U$, where h^U is the maximum allowable value of the dynamic pressure, $h(\boldsymbol{r},\boldsymbol{v}) = \rho(r)\boldsymbol{v}\cdot\boldsymbol{v}/2$. Thus, the continuous-valued state spaces, that is, the so-called invariant sets [see Eq. (6)] are given by

$$\mathbb{X}(q) = \begin{cases} \left\{[\boldsymbol{r},\boldsymbol{v},m] \in \mathbb{R}^7 : 0 \le h(\boldsymbol{r},\boldsymbol{v}) \le h_2^U\right\} \\ \qquad\qquad \text{if} \quad q \in \{q_a, q_b, q_g\} \\ \left\{[\boldsymbol{r}_A,\boldsymbol{r}_R,\boldsymbol{v}_A,\boldsymbol{v}_R,m_A,m_R] \in \mathbb{R}^{14} : \right. \\ \quad \left. 0 \le h(\boldsymbol{r}_A,\boldsymbol{v}_A) \le h_1^U,\ 0 \le h(\boldsymbol{r}_R,\boldsymbol{v}_R) \le h_1^U\right\} \\ \qquad\qquad \text{if} \quad q \in \{q_c, q_d\} \\ \left\{[\boldsymbol{r},\boldsymbol{v},m] \in \mathbb{R}^7 : 0 \le h(\boldsymbol{r},\boldsymbol{v}) \le h_1^U\right\} \\ \qquad\qquad \text{if} \quad q \in \{q_e, q_f\} \end{cases}$$

where the numerical subscripts on h^U indicate different maximum values of dynamic pressure. Because the controls are the thrust attitude direction cosines, that is, elements of S^2, we have

$$\mathbb{U}(q) = \begin{cases} \{\boldsymbol{u} \in \mathbb{R}^3 : |\boldsymbol{u}| = 1\} & \text{if} \quad q \in \{q_a, q_b, q_e, q_f, q_g\} \\ \{\boldsymbol{u}_A, \boldsymbol{u}_R \in \mathbb{R}^3 : |\boldsymbol{u}_A| = 1, |\boldsymbol{u}_R| = 1\} & \text{if} \quad q \in \{q_c, q_d\} \end{cases}$$

E. Event Sets

In the following subsections, we enumerate and define the various event sets $\mathbb{E}(q,q')$. For the purpose of brevity, we do not write these sets explicitly in terms of the event functions and their lower and upper bounds because such a representation should be quite obvious from the context. For example, when a condition is stated as an equality, it is apparent that the lower and upper bounds on the corresponding event function are equal. An enumeration of the switching sets is facilitated by the adjacency matrix [cf. Eq. (12)]

$$\begin{array}{c} \\ q_a \\ q_b \\ q_c \\ q_d \\ q_e \\ q_f \\ q_g \end{array} \overbrace{\begin{array}{c} \begin{array}{ccccccc} q_a & q_b & q_c & q_d & q_e & q_f & q_g \end{array} \\ \left(\begin{array}{ccccccc} 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \end{array}\right) \end{array}}^{} = \boldsymbol{A} \quad (47)$$

which follows quite simply from a combination of the physics of the problem[44] and by CONOPS (also see the Appendix). By definition, the switching sets for which the elements of $\boldsymbol{A}$ are zero are all empty sets. The switching sets corresponding to the nonzero elements of $\boldsymbol{A}$ are as follows:

$$\begin{array}{lccc} \text{Switching sets:} & \mathcal{S}(q_a,q_b), & \mathcal{S}(q_a,q_g), & \mathcal{S}(q_b,q_a) \\ & \mathcal{S}(q_d,q_c), & \mathcal{S}(q_e,q_f), & \mathcal{S}(q_b,q_g) \\ & \mathcal{S}(q_g,q_a), & \mathcal{S}(q_f,q_e) & \end{array}$$

From the definitions of the discrete states, Eq. (29), it follows that, for $q, q' \in \{q_a, q_b, q_e, q_f, q_g\}$, we have $\boldsymbol{x} = [\boldsymbol{r},\boldsymbol{v},m] \in \mathbb{R}^7$, $\boldsymbol{u} \in \mathbb{R}^3$, $\boldsymbol{x}' = [\boldsymbol{r}',\boldsymbol{v}',m'] \in \mathbb{R}^7$, and $\boldsymbol{u}' \in \mathbb{R}^3$. Similarly, for $q, q' \in \{q_c, q_d\}$, we have $\boldsymbol{x} = [\boldsymbol{r}_A,\boldsymbol{r}_R,\boldsymbol{v}_A,\boldsymbol{v}_R,m_A,m_R] \in \mathbb{R}^{14}$ and $\boldsymbol{u} = [\boldsymbol{u}_A,\boldsymbol{u}_R] \in \mathbb{R}^6$ and $\boldsymbol{x}' = [\boldsymbol{r}'_A,\boldsymbol{r}'_R,\boldsymbol{v}'_A,\boldsymbol{v}'_R,m'_A,m'_R] \in \mathbb{R}^{14}$ and $\boldsymbol{u}' = [\boldsymbol{u}_A,\boldsymbol{u}_R] \in \mathbb{R}^6$. In all of these cases, the switching sets are given quite simply by the continuity conditions, that is [see Eq. (13)],

$$\mathcal{S}(q,q') = \mathcal{C}(q,q') \quad (48)$$

$$\begin{array}{lccc} \text{Switching sets:} & \mathcal{S}(q_a,q_c), & \mathcal{S}(q_a,q_d), & \mathcal{S}(q_b,q_c) \\ & \mathcal{S}(q_b,q_d), & \mathcal{S}(q_g,q_c), & \mathcal{S}(q_g,q_d) \end{array}$$

Let $\tau = \tau' = t_e$; then, all of these switching sets are defined in terms of the following event conditions:

$$\boldsymbol{r}(t_e) = \boldsymbol{r}_A(t_e) = \boldsymbol{r}_R(t_e) \quad (49)$$

$$\boldsymbol{v}(t_e) = \boldsymbol{v}_A(t_e) = \boldsymbol{v}_R(t_e) \quad (50)$$

$$m(t_e) = m_A(t_e) + m_R(t_e) \quad (51)$$

$$M_R \le m_R(t_e) \le M_0 \quad (52)$$

where M_R is the dry mass of the return vehicle and M_0 is the gross lift-off mass.

$$\text{Switching sets:} \quad \mathcal{S}(q_c,q_f), \qquad \mathcal{S}(q_d,q_e), \qquad \mathcal{S}(q_d,q_f)$$

Let $\boldsymbol{x} = [\boldsymbol{r},\boldsymbol{v},m] \in \mathbb{R}^7$ for $q \in \{q_e, q_f\}$ and $\tau = \tau' = t_e$; then,

$$\boldsymbol{r}(t_e) \in \{\boldsymbol{r}_A(t_e), \boldsymbol{r}_R(t_e)\} \quad (53)$$

$$\boldsymbol{v}(t_e) \in \{\boldsymbol{v}_A(t_e), \boldsymbol{v}_R(t_e)\} \quad (54)$$

$$m(t_e) \in \{m_A(t_e), m_R(t_e)\} \quad (55)$$

$$\text{Endpoint set:} \quad \mathcal{E}(q^0, q^{N_s})$$

The endpoint set is a nonswitching event set that represents the boundary conditions. The boundary conditions on the discrete state are given by

$$q^0 \in \mathcal{Q}_0 := \{q_a, q_b\} \qquad q^{N_s} \in \mathcal{Q}_f := \{q_c, q_e\} \quad (56)$$

Let $\boldsymbol{x} = [\boldsymbol{r}(t_0), \boldsymbol{v}(t_0), m(t_0)] \in \mathbb{R}^7$ and $\boldsymbol{u} \in \mathbb{R}^3$ and $\boldsymbol{x}' = [\boldsymbol{r}_A(t_f), \boldsymbol{r}_R(t_f), \boldsymbol{v}_A(t_f), \boldsymbol{v}_R(t_f), m_A(t_f), m_R(t_f)] \in \mathbb{R}^{14}$ and $\boldsymbol{u}' = [\boldsymbol{u}_A, \boldsymbol{u}_R] \in \mathbb{R}^6$. The initial conditions on the continuous-valued state can be specified in terms of a launch pad longitude α_l and latitude δ_l:

$$\boldsymbol{r}(t_0) = R_\oplus {}^N\boldsymbol{C}^E(t_0)[\cos\alpha_l \cos\delta_l, \sin\alpha_l \cos\delta_l, \sin\delta_l]^T \quad (57)$$

$$\boldsymbol{v}(t_0) = \boldsymbol{\Omega}_\oplus \times \boldsymbol{r}(t_0) \quad (58)$$

$$m(t_0) = M_0 \quad (59)$$

where ${}^N\boldsymbol{C}^E(t_0) \in SO(3)$ is the coordinate transformation matrix between the Earth-centered, Earth-fixed system E and the Earth-centered-inertial system N. Similarly, the final-time conditions on

the return vehicle is specified as

$$\boldsymbol{r}_R(t_f) = R_{\oplus}{}^N\boldsymbol{C}^E(t_f)[\cos\alpha_r\cos\delta_r, \sin\alpha_r\cos\delta_r, \sin\delta_r]^T \quad (60)$$

$$\alpha^L \le \alpha_r \le \alpha^U \quad (61)$$

$$\delta^L \le \delta_r \le \delta^U \quad (62)$$

When $\boldsymbol{v}(t_f)$ is allowed to be free and when the latitude and longitude are chosen to be an appropriate point or region on or above the surface of the Earth, this condition is essentially a splashdown constraint[47] for water recovery or a parachute drop point for land recovery. Of course, we could impose hard conditions on $\boldsymbol{v}(t_f)$ as well, to model a return to some region in state space [such as a terminal area energy management (TAEM) point]. Because our example problem does not model lift, we limit our attention to a simple return constraint.

The end conditions on the ascent vehicle are specified in terms of the classical orbital elements $\{a, e, i, \Omega, \omega\}$ that can be expressed in terms of the state variables as[50]

$$a(\boldsymbol{r}, \boldsymbol{v}) = \frac{|\boldsymbol{r}|}{2 - |\boldsymbol{r}||\boldsymbol{v}|^2/\mu} \quad (63)$$

$$e(\boldsymbol{r}, \boldsymbol{v}) = |\boldsymbol{e}(\boldsymbol{r}, \boldsymbol{v})| \quad (64)$$

$$\sin i(\boldsymbol{r}, \boldsymbol{v}) = |\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v})| \quad (65)$$

$$\cos\Omega(\boldsymbol{r}, \boldsymbol{v}) = \boldsymbol{i}_x \cdot \frac{\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v})}{|\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v})|} \quad (66)$$

$$\cos\omega(\boldsymbol{r}, \boldsymbol{v}) = \frac{\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v}) \cdot \boldsymbol{e}(\boldsymbol{r}, \boldsymbol{v})}{|\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v})||\boldsymbol{e}(\boldsymbol{r}, \boldsymbol{v})|} \quad (67)$$

where $\boldsymbol{i}_x$ and $\boldsymbol{i}_z$ are the unit vectors along the vernal equinox and the spin axis of the Earth, respectively, and $\boldsymbol{e}(\boldsymbol{r}, \boldsymbol{v})$, and $\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v})$ are the eccentricity and nodal vector functions, respectively, defined as

$$\boldsymbol{e}(\boldsymbol{r}, \boldsymbol{v}) = \left(|\boldsymbol{v}|^2 - \frac{\mu}{|\boldsymbol{r}|}\right)\boldsymbol{r} - (\boldsymbol{r} \cdot \boldsymbol{v})\boldsymbol{v} \quad (68)$$

$$\boldsymbol{n}(\boldsymbol{r}, \boldsymbol{v}) = \frac{\boldsymbol{i}_z \times (\boldsymbol{r} \times \boldsymbol{v})}{|\boldsymbol{r} \times \boldsymbol{v}|} \quad (69)$$

Thus, the event conditions for the ascent vehicle are expressed in terms of a set of target values for $a(\boldsymbol{r}_A(t_f), \boldsymbol{v}_A(t_f))$, $\boldsymbol{e}(\boldsymbol{r}_A(t_f), \boldsymbol{v}_A(t_f))$, etc.

F. Flight Plan Automaton

We order the elements of $\mathcal{Q}$ as

$$[Q] = [q_a, q_b, q_c, q_d, q_e, q_f, q_g]$$

Hence, $\mathcal{Q}_0$ and $\mathcal{Q}_f$ are transcribed as [cf. Eq. (56) and Sec. III]

$$A^0 := \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \qquad A^f := \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad (70)$$

By incorporating Eq. (70) with Eq. (47), we can now construct the discrete control space $\mathbb{U}_D$ according to Eq. (27). Thus, problem FIP can be quickly solved to yield a variety of flight plans. As an example, a possible flight plan is given by the sequence $\boldsymbol{q} = (q_a, q_b, q_d, q_c)$, corresponding to a wait at the launch pad for a launch window, a launch of the mated system, vehicle separation to the two-agent mode q_d, and a terminating flight of both vehicles coasting to their respective target sets. Obviously, other flight plans are possible. Based on the discussions of Sec. III, it is apparent that the totality of flight plans is less than $7^{(N_s+1)}$. A variety of these plans are described in Ref. 44. Thus, once the discrete state space $\mathcal{Q}$ and the associated digraph is defined, CONOPS is simply a flight plan described in terms of a walk.

G. Cost Functions

As indicated in Ref. 51, fuel expenditures dictate the feasibility of space missions. The propellant cost function may be expressed either in the generalized Lagrange form or a Mayer form (see Ref. 23). For brevity, we describe only the Lagrange form. Following Ref. 51, we take the L^1 norm of the thrust function as a direct measure of the total fuel expended; hence, we can write the running cost as

$$F(\boldsymbol{x}, \boldsymbol{u}, q) = \begin{cases} |T(q)| & \text{if} \quad q \in \{q_a, q_b, q_e, q_f, q_g\} \\ |T_A(q)| + |T_R(q)| & \text{if} \quad q \in \{q_c, q_d\} \end{cases} \quad (71)$$

There is no cost to switch from one location to another; that is, the digraph is unweighted. In addition, we have no cost associated with the end points; hence, we have

$$E(\cdot, q, q') = 0 \qquad \forall q, q' \in \mathcal{Q} \quad (72)$$

Thus, this is a pure Lagrange form formulation.

V. Numerical Example

Tables 1 and 2 provide the appropriate numerical data for the hybrid system. These numbers are based on a generic two-agent launch vehicle with the launch point being the NASA Kennedy Space Center, Florida. The target low-Earth-orbit parameters for the ascent vehicle, and the return point constraint for the return vehicle, are specified in Table 2. The return point corresponds to an arbitrarily chosen region in the Atlantic Ocean. The hybrid system trajectories for various flight plans are discussed in Refs. 23 and 44. The numerical results were obtained using DIDO,[38] a MATLAB

Table 1 Data for vehicle and environmental model

Parameter	Value	Unit
$C_{D,1}$	0.20	——
$C_{D,2}$	0.25	——
H_s	7,254	m
M_0	55,000	kg
M_R	1,500	kg
$R_\oplus$	6,378	km
S_1	1.76	m^2
S_2	2.18	m^2
T_b	1.80×10^6	N
T_d	1.33×10^5	N
T_e	1.33×10^5	N
T_g	1.33×10^5	N
γ_b	576	kg/s
γ_d	32	kg/s
γ_e	32	kg/s
γ_g	32	kg/s
μ	3.986×10^5	km^3/s^2
ρ_0	1.225	kg/m^3
$\Omega_\oplus$	2π	/day

Table 2 Data for endpoint conditions

Parameter	Value	Unit
$a[t_f]$	7016	km
$e[t_f]$	0.025	——
$i[t_f]$	28.50	deg
α_l	−80.55	deg E
α^L	−73.58	deg E
α^U	−73.56	deg E
δ_l	28.50	deg N
δ^L	28.23	deg N
δ^U	28.25	deg N
$\omega[t_f]$	free	——
$\Omega[t_f]$	free	——

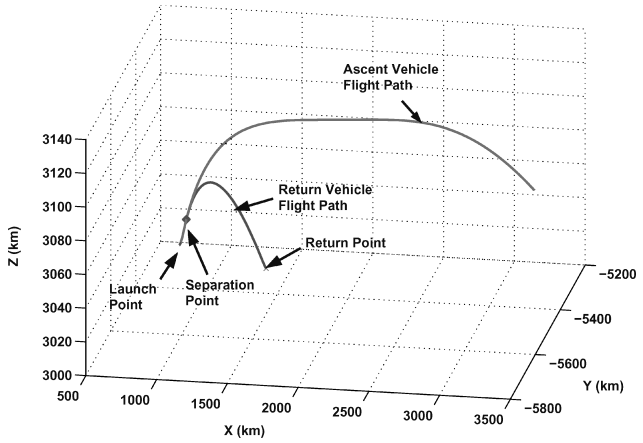

Fig. 8 Optimized flight path of two-agent system.

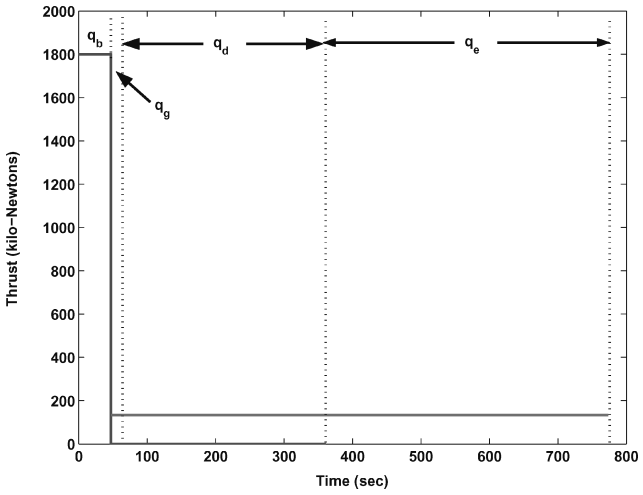

Fig. 9 Thrusting modes for two-agent launch system.

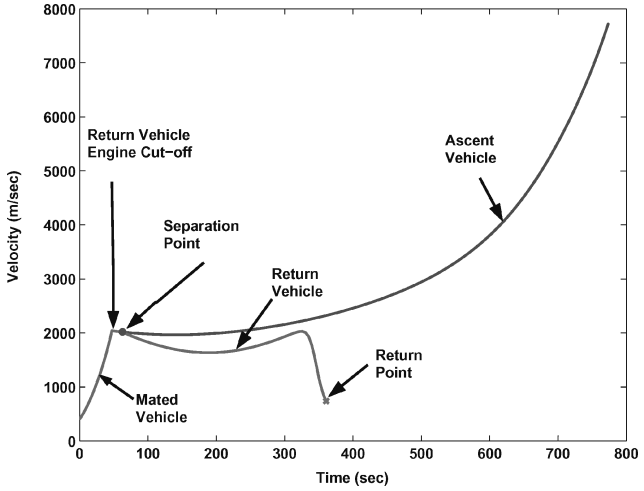

Fig. 10 Velocity magnitudes of two-agent launch system.

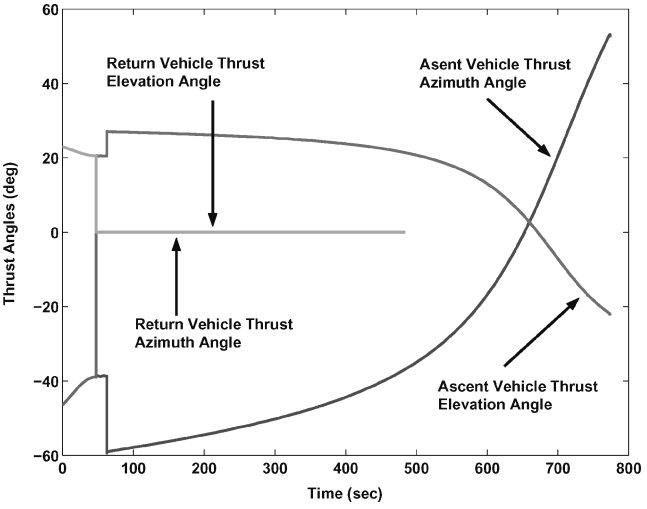

Fig. 11 Thrust attitude angles (controls) for two-agent launch system.

application package that extensively uses the TOMLAB[52] solver suite to incorporate many of the ideas described in Sec. III.

In limiting the scope of this section, we discuss the main features of a four-mode flight plan,

$$\boldsymbol{q}_4 := (q_b, q_g, q_d, q_e) \tag{73}$$

The optimized flight path of the two-agent hybrid system is shown in Fig. 8. Corresponding to the switches, we have an event condition given by a continuity condition $\mathcal{S}(q_b, q_g)$, a switching condition corresponding to a separation condition $\mathcal{S}(q_g, q_d)$, followed by a dimensional switching condition $\mathcal{S}(q_d, q_e)$. The thrust and velocity magnitude shown in Figs. 9 and 10 show the salient points of this hybrid structure.

The thrust attitude angles are, for elevation

$$\sin^{-1}(u_3) \tag{74}$$

and for azimuth

$$\tan^{-1}(u_2/u_1) \tag{75}$$

and are much more informative than the control parameters (direction cosines), as shown in Fig. 11. Note that the PS knotting method successfully represents the switches and corners in all of these plots.

As part of the validation procedure, the trajectories obtained from the knotting method are investigated with regard to feasibility. Following Refs. 38 and 53, we define feasibility in this context as whether or not the control history generated from DIDO, if propagated via an integrator, would achieve the desired final conditions (as well as match the states produced from DIDO). Using piecewise cubic Hermite interpolation for the controls, the initial conditions were propagated using ode45 in MATLAB. The L^∞ error norms between the propagated solutions and the DIDO solutions were 45 m in position and 0.003 m/s in velocity. Because the difference between the propagated and DIDO results are not visible in a plot, only the DIDO solutions are shown in Figs. 8–11.

All of the numerical results were obtained quite easily from infeasible guesses. A typical two-agent problem takes about 20 min to solve on a 2.4-GHz Pentium IV machine running Windows NT and MATLAB 6.5. As described in Refs. 8 and 54, there are a number of ways to reduce the run time; none of these options were used in this preliminary study. Based on a combination of theoretical and computational analysis, it has been shown in Refs. 8 and 55 that at least an order of magnitude reduction in run time is possible, even with random initial guesses and current processor technology.

VI. Conclusions

The bold complexity of upcoming missions in both the civilian and military sectors has challenged engineers to increase autonomy as a means to manage complexity. Because space missions are strongly driven by optimality considerations, it is crucial to solve the outer-loop control problems within the context of optimization theory. The emerging language of hybrid optimal control theory provides a formal means to approach these problems. Given that we are now able to solve a wide class of ordinary optimal control problems routinely, it is possible to articulate a promising methodology for solving hybrid optimal control problems. In using the efficiency of PS methods, it is quite straightforward to show that

hybrid optimal control problems can be discretized to MVP problems. Because MVPs are NP hard, it is necessary to explore ideas that reduce the combinational complexity of the problem. The feasible integer programming problem proposed in this paper provides a direction toward this goal. A vast number of important research questions remain open. Answering these questions requires one to straddle multiple disciplines: discrete mathematics, control theory, optimization, approximation theory, and the ever-important intangible of translating engineering requirements to mathematical conditions. There is no doubt that hybrid problems will continue to challenge us in the years to come.

Appendix: Discrete Modeling

When a problem can be naturally posed as an ordinary optimal control problem, it can be solved quite readily by modern methods, even when the solution has switches.[51] Although it is generally inadvisable to convert an ordinary optimal control problem to a hybrid one, such a conversion can be done in a straightforward manner. On the other hand, when a problem is fundamentally hybrid, questions about modeling the discrete system take center stage. In this Appendix, we demonstrate how complex discrete models can be built by an elementary process of taking Cartesian products of simpler, intuitive, discrete spaces. Although one of the main utilities of Cartesian products is exactly that of building complex spaces from simpler ones, we also show here how to perform certain quick reductions in discrete spaces. We use the two-agent launch problem to illustrate the concepts.

The discrete (categorical) space representing the information on the state of the mating can be defined as

$$\mathcal{Q}_1 := \{\text{mated, separated}\}$$

If the discrete state space had just two elements, and the continuous state space $\mathbb{X}(q)$ were unconstrained, then we could write

$$\mathbb{X}(\text{mated}) = \mathbb{R}^7, \qquad \mathbb{X}(\text{separated}) = \mathbb{R}^{14}$$

Note that much of the state spaces discussed in the computational hybrid control literature have significantly fewer dimensions than 14. Because we allow vehicles to enter and leave the continuous state space (dimensional switching), the discrete state space corresponding to the vehicles under consideration can be written as follows for ascent vehicles (AV), return vehicles (RV), and both vehicles (BV):

$$\mathcal{Q}_2 := \{\text{AV, RV, BV}\}$$

In examining the Cartesian product,

$$\mathcal{Q}_1 \times \mathcal{Q}_2$$

it is clear that the elements

$$\begin{pmatrix}\text{mated}\\ \text{AV}\end{pmatrix}, \qquad \begin{pmatrix}\text{mated}\\ \text{RV}\end{pmatrix} \tag{A1}$$

have no physical meaning. Thus, out of a possible set of six elements of the product space, we can immediately rule out two values as physically impossible. For the purposes of simplicity, we make no distinction between the AV and RV characteristics. This simplification, which can be considered part of the CONOPS, allows us to write

$$\underbrace{\begin{pmatrix}\text{separated}\\ \text{AV}\end{pmatrix}}_{\text{single}} = \underbrace{\begin{pmatrix}\text{separated}\\ \text{RV}\end{pmatrix}}_{\text{single}} \tag{A2}$$

so that the total number of allowable valuations of q is reduced to three, with the remainder of the valuations given by

$$\underbrace{\begin{pmatrix}\text{mated}\\ \text{BV}\end{pmatrix}}_{\text{mated}}, \qquad \underbrace{\begin{pmatrix}\text{separated}\\ \text{BV}\end{pmatrix}}_{\text{separated}} \tag{A3}$$

Thus, we define a new set,

$$\mathcal{Q}_{12} := \{\text{mated, separated, single}\}$$

where single is distinguished from separated in that the other vehicle has not entered the continuous state space $\mathbb{X}(q)$. Thus, Q_{12} accounts for physics and CONOPS. A modeling based on physics alone is described in Ref. 44, wherein it is shown that the problem has a discrete state space of cardinality 12. In any case, the discrete state space that represents the state of thrusting of the solid rockets is given by

$$\mathcal{Q}_3 = \{\text{on, off}\}$$

If the engines were based on liquid propellants, defining $\mathcal{Q}_3$ would be unnecessary because thrusting would be part of the continuous control space $\mathbb{U}(q)$. For solid rockets, the modeling is naturally discrete. When the vehicles contain both liquid-based and solid-propellant engines, then the thrust controller has both continuous and discrete elements. Finally, the discrete space corresponding to the number of engines in the hybrid system is

$$\mathcal{Q}_4 = \{1, 2\}$$

where we assume that each vehicle has just one engine.

In a cursory investigation of the Cartesian product

$$\mathcal{Q}_{12} \times \mathcal{Q}_3 \times \mathcal{Q}_4$$

it appears that the cardinality of the discrete state space is $3 \times 2 \times 2 = 12$. This is not true because two of the valuations are not physically possible. That is, from $\mathcal{Q}_{12} \times \mathcal{Q}_4$, the pair (single, 2) is impossible; thus, with two associations with it, on and off, not being possible, the cardinality of the possible state space is reduced to 10. Thus, we have

$$\mathcal{Q} \subset \mathcal{Q}_{12} \times \mathcal{Q}_3 \times \mathcal{Q}_4$$

We further constrain the discrete state space by equating the following two possibilities:

$$\underbrace{\begin{pmatrix}\text{mated}\\ \text{on}\\ 1\end{pmatrix}}_{\begin{pmatrix}\text{mated}\\ \text{on/off}\end{pmatrix}} = \underbrace{\begin{pmatrix}\text{mated}\\ \text{off}\\ 1\end{pmatrix}}_{\begin{pmatrix}\text{mated}\\ \text{on/off}\end{pmatrix}} \tag{A4}$$

corresponding to either one of the engines in the mated case being off. Carrying out such analyses, it is possible to articulate a discrete state space Q represented in Eq. (28). Further details on modeling, including a description of the full 12 states and the corresponding adjacency matrices, are described in Ref. 44.

Acknowledgments

We thank Steven E. Matousek, Jon A. Sims, and Stacy Weinstein, all of the Jet Propulsion Laboratory, California Institute of Technology, for introducing us to the challenges of upcoming space mission planning problems. The asteroid example presented in Sec. I of this paper is a result of the conversations we had with Jon Sims and Stacy Weinstein. We thank Timothy J. Brand of the Charles Stark Draper Laboratories, Inc., for introducing us to the complexities of the two-agent launch problem discussed in Sec. IV. We also thank John P. Riehl for his enthusiastic support of pseudospectral methods for solving mission planning problems for NASA. We express our particular appreciation for Ping Lu, whose incisive comments greatly helped in strengthening this paper. For lack of space, we do not list the numerous personnel from the U.S. Navy and Marine Corps, and the U.S. Air Force, who spent their precious time outlining to us the enormous technical challenges facing the U.S. military. We thank them most of all.

References

[1]Betts, J. T., *Practical Methods for Optimal Control Using Nonlinear Programming*, Advances in Control and Design Series, Society for Industrial and Applied Mathematics, Philadelphia, 2001.

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

[3]Ross, I. M., and Fahroo, F., "A Perspective on Methods for Trajectory Optimization," AIAA Paper 2002-4727, Aug. 2002.

[4]Pesch, H. J., "A Practical Guide to the Solution of Real-Life Optimal Control Problems," *Control and Cybernetics*, Vol. 23, No. 1/2, 1994, pp. 7–60.

[5]Ross, I. M., and Fahroo, F., "Pseudospectral Knotting Methods for Solving Optimal Control Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 3, 2004, pp. 397–405.

[6]Warner, M. S., and Hodges, D. H., "Solving Optimal Control Problems Using *hp*-Version Finite Elements in Time," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 1, 2000, pp. 86–94.

[7]Milam, M. B., Mushambi, K., and Murray, R. M., "A New Computational Approach to Real-Time Trajectory Generation for Constrained Mechanical Systems," *Proceedings of the IEEE Conference on Decision and Control*, IEEE Press, Piscataway, NJ, 2000.

[8]Ross, I. M., and Fahroo, F., "Issues in the Real-Time Computation of Optimal Control," *Mathematical and Computer Modelling* (to be published).

[9]Audet, C., and Dennis, J. E., "Pattern Search Algorithms For Mixed Variable Programming," *SIAM Journal of Optimization*, Vol. 11, No. 3, 2000, pp. 573, 594.

[10]Lucidi, S., Piccialli, V., and Sciandrone, M., "An Algorithm Model For Mixed Variable Programming," D.I.S. Technical Rept., 17-02, Univ. of Rome "La Sapienza," Rome, 2002.

[11]Kreyzig, E., *Advanced Engineering Mathematics*, 7th ed., Wiley, New York, 1993, Chap. 22.

[12]*Stateflow and Stateflow Coder User's Guide*, MathWorks, Inc., Natick, MA, 2004.

[13]Burkard, R. E., Deĭneko, V. G., Van Dal, R., Van der Veen, J. A. A., and Woeginger, G. J., "Well-Solvable Special Cases of the Traveling Salesman Problem: A Survey," *SIAM Review*, Vol. 40, No. 3, 1998, pp. 496–546.

[14]von Stryk, O., and Glocker, M., "Numerical Mixed-Integer Optimal Control and Motorized Traveling Salesmen Problems," *APII Journal Europeen des Systèmes Automatises [European Journal of Control]*, Vol. 35, No. 4, 2001, pp. 519–533.

[15]von Stryk, O., and Glocker, M., "Decomposition of Mixed-Integer Optimal Control Problems Using Branch and Bound and Sparse Direct Collocation," *Proceedings of ADPM 2000. The 4th International Conference on Automation of Mixed Processes: Hybrid Dynamic Systems*, Shaker, Aachen, Germany, 2000, pp. 99–104.

[16]Glocker, M., and von Stryk, O., "Hybrid Optimal Control of Motorized Traveling Salesmen and Beyond," *Proceedings of the 15th IFAC World Congress on Automatic Control*, IFAC, Barcelona, July 2002.

[17]Fowler, W. T., and Gottlieb, R. G., "A Result Applicable to a Particular Class of Branched Optimization Problems," *Engineering Optimization*, Vol. 2, 1977, pp. 223–337.

[18]Sussmann, H. J., "A Maximum Principle for Hybrid Optimal Control Problems," *Proceedings of the 38th IEEE Conference on Decision and Control*, IEEE Press, Piscataway, NJ, Dec. 1999.

[19]Sussmann, H. J., "A Nonsmooth Hybrid Maximum Principle," *Lecture Notes in Control and Information Sciences*, Vol. 246, Springer-Verlag, New York, 1999, pp. 325–354.

[20]Sussmann, H. J., "Set-Valued Differentials and the Hybrid Maximum Principle," *Proceedings of the 39th IEEE Conference on Decision and Control*, IEEE Press, Piscataway, NJ, 2000.

[21]Ross, I. M., and Fahroo, F., "A Direct Method for Solving Nonsmooth Optimal Control Problems," *Proceedings of the 2002 World Congress of the International Federation on Automatic Control*, IFAC, Barcelona, July 2002.

[22]Ross, I. M., and Fahroo, F., "Discrete Verification of Necessary Conditions for Switched Nonlinear Optimal Control Systems," *Proceedings of the American Control Conference*, Boston, June 2004.

[23]Ross, I. M., and D'Souza, C. N., "Rapid Trajectory Optimization of Multi-Agent Hybrid Systems," AIAA Paper 2004-5422, Aug. 2004.

[24]Engell, S., Frehse, G., and Schnieder, E. (eds.), *Modelling, Analysis and Design of Hybrid Systems*, Lecture Notes in Control and Information Sciences, Vol. 279, Springer-Verlag, Berlin, 2002.

[25]Antsaklis, P. J., and Nerode, A., "Guest Editorial Hybrid Control Systems: An Introductory Discussion to the Special Issue," *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, 1998, pp. 457–460.

[26]Lygeros, J., Johansson, K. H., Simic, S. N., Zhang, J., and Sastry, S. S., "Dynamical Properties of Hybrid Automata," *IEEE Transactions on Automatic Control*, Vol. 48, No. 1, 2003, pp. 2–17.

[27]Branicky, M. S., Borkar, V. S., and Mitter, S. K., "A Unified Framework for Hybrid Control: Model and Optimal Control Theory," *IEEE Transactions on Automatic Control*, Vol. 43, No. 1, 1998, pp. 31–45.

[28]Buss, M., Glocker, M., Hardt, M., Stryk, O., Bulirsch, R., and Schmidt, G., "Nonlinear Hybrid Dynamical Systems: Modeling, Optimal Control and Applications," *Lecture Notes in Control and Information Sciences*, Vol. 279, Springer-Verlag, New York, 2002, pp. 311–335.

[29]Buss, M., Hardt, M., and von Stryk, O., "Numerical Solution of Hybrid Optimal Control Problems with Applications in Robotics," *Proceedings of the 15th IFAC World Congress on Automatic Control*, IFAC, Barcelona, July 2002.

[30]Zhang, J., Johansson, K. H., Lygeros, J., and Sastry, S., "Zeno Hybrid Systems," *International Journal of Robust and Nonlinear Control*, Vol. 11, 2001, pp. 435–451.

[31]Kolmogorov, A. N., and Fomin, S. V., *Elements of the Theory of Functions and Functional Analysis*, Dover, Mineola, NY, 1999.

[32]Vinter, R. B., *Optimal Control*, Birkhäuser, Boston, 2000.

[33]Hartl, R. F., Sethi, S. P., and Vickson, R. G., "A Survey of the Maximum Principles for Optimal Control Problems with State Constraints," *SIAM Review*, Vol. 37, No. 2, 1995, pp. 181–218.

[34]Stevens, R., and Ross, I. M., "Preliminary Design of Earth–Mars Cyclers Using Solar Sails," *Journal of Spacecraft and Rockets*, Vol. 42, No. 1, 2005, pp. 132–137; also American Astronautical Society, Paper AAS 03-244, Feb. 2003.

[35]Stevens, R., Ross, I. M., and Matousek, S. E., "Earth–Mars Return Trajectories Using Solar Sails," *55th International Astronautical Congress*, Vancouver, Paper IAC-04-A.2.08, Oct. 2004.

[36]Xu, X., and Antsaklis, P. J., "Results and Perspectives on Computational Methods for Optimal Control of Switched Systems," *Proceedings of the Sixth International Workshop on Hybrid Systems: Computation and Control (HSCC 2003)*, Prague, April 2003.

[37]Bardi, M., and Capuzzo-Dolcetta, I., *Optimal Control and Viscosity Solutions of Hamilton–Jacobi–Bellman Equations*, Birkhäuser, Boston, 1997.

[38]Ross, I. M., and Fahroo, F., "User's Manual for DIDO 2002: A MATLAB Application Package for Dynamic Optimization," NPS Technical Rept. AA-02-002, Dept. of Aeronautics and Astronautics, Naval Postgraduate School, Monterey, CA, June 2002.

[39]Elnagar, J., Kazemi, M. A., and Razzaghi, M., "The Pseudospectral Legendre Method for Discretizing Optimal Control Problems," *IEEE Transactions on Automatic Control*, Vol. 40, No. 10, 1995, pp. 1793–1796.

[40]Ross, I. M., and Fahroo, F., "Legendre Pseudospectral Approximations of Optimal Control Problems," *Lecture Notes in Control and Information Sciences*, Vol. 295, Springer-Verlag, New York, 2003, pp. 327–342.

[41]Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods in Fluid Dynamics*, Springer-Verlag, New York, 1988.

[42]Gong, Q., Ross, I. M., Kang, W., and Fahroo, F., "Convergence of Pseudospectral Methods for Constrained Nonlinear Optimal Control Problems," *Intelligent Systems and Control*, Series on Modelling, Identification and Control, Acta, Calgary, AB, Canada, 2004.

[43]Lu, P., Sun, S., and Tsai, B., "Closed-Loop Endoatmospheric Ascent Guidance," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 2, 2003, pp. 283–294.

[44]Ross, I. M., and D'Souza, C. N., "Discrete Modeling and Flight Plan Automata for Multi-Agent Launch Systems," NPS Technical Rept., NPS-MAE-04-006, Dept. of Mechanical and Astronautical Engineering, Naval Postgraduate School, Monterey, CA, Sept. 2004.

[45]Ross, I. M., King, J. T., and Fahroo, F., "Designing Optimal Spacecraft Formations," AIAA Paper 2002-4635, Aug. 2002.

[46]Infeld, S., Josselyn, S., Murray, W., and Ross, I. M., "Design and Control of Libration Point Spacecraft Formations," AIAA Paper 2004-4786, 2004.

[47]Weigel, N., and Well, K. H., "Dual Payload Ascent Trajectory Optimization with a Splash-Down Constraint," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 1, 2000, pp. 45–52.

[48]Ross, I. M., D'Souza, C. N., Fahroo, F., and Ross, J. B., "A Fast Approach to Multi-Stage Launch Vehicle Trajectory Optimization," AIAA Paper 2003-5639, Aug. 2003.

[49]Rea, J., "Launch Vehicle Trajectory Optimization Using a Legendre Pseudospectral Method," AIAA Paper 2003-5640, Aug. 2003.

[50]Prussing, J. E., and Conway, B. A., *Orbital Mechanics*, Oxford Univ. Press, New York, 1993.

[51]Ross, I. M., "How to Find Minimum-Fuel Controllers," AIAA Paper 2004-5346, Aug. 2004.

[52]Holmström, K., Göran, A. O., and Edvall, M. M., "User's Guide for Tomlab 4.0.6," Tomlab Optimization, Vasteras, Sweden, Aug. 2003.

[53]Josselyn, S., and Ross, I. M., "Rapid Verification Method for the Trajectory Optimization of Reentry Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 3, 2003, pp. 505–508.

[54]Strizzi, J., Ross, I. M., and Fahroo, F., "Towards Real-Time Computation of Optimal Controls for Nonlinear Systems," AIAA Paper 2002-4945, Aug. 2002.

[55]Ross, I. M., Fahroo, F., and Gong, Q., "A Spectral Algorithm for Pseudospectral Methods in Optimal Control," *Proceedings of the 10th International Conference on Cybernetics and Information Technologies, Systems and Applications (CITSA)*, CITSA, Orlando, FL, 2004, pp. 104–109.