Finite Set Control Transcription for Optimal Control Applications

Stuart A. Stanton* and Belinda G. Marchand†
University of Texas at Austin, Austin, Texas 78712-0235

DOI: 10.2514/1.44056

Earlier studies focus on the development of the finite set control transcription method for the treatment of a special class of optimal control problems. Specifically, problems with multiple control variables for which each, or some, are independently restricted to a finite set of values. At some level, even continuous control problems can be modeled within the context of finite set control. To demonstrate the possible range of applications of this methodology, on–off and variable actuation schemes are considered here. The goal is to demonstrate the utility and versatility of the finite set control transcription method. The ensuing solutions are characterized as optimal switching schedules between feasible control values. The finite set control transcription allows control switches to be determined over a continuous time spectrum, overcoming many of the limitations associated with traditionally discretized solutions. The method is particularly well suited for problems involving multiple but independently switching control parameters. Specifically, a significant increase in computational efficiency is achieved, in contrast to existing methods, due to the reduced dimensionality of the parameter optimization problem.

Nomenclature

A_t	=	throat area, m^2
$\boldsymbol{a}$	=	eigenvector real part
$\boldsymbol{C}$	=	direction cosine matrix
$\boldsymbol{c}$	=	constraint vector
c	=	exhaust velocity, m/s
c^*	=	characteristic velocity, m/s
$\boldsymbol{d}$	=	valve core rod position vector, mm
F	=	cost function
F_t	=	thrust, N
$\boldsymbol{f}$	=	dynamics function
g	=	gravitational constant, m/s^2
$\boldsymbol{J}$	=	inertia tensor matrix
$\mathcal{J}$	=	cost functional
$\boldsymbol{L}$	=	diagonal dimension matrix
L	=	integrand cost
l	=	length, m
m	=	number of feasible control values
$\bar{m}$	=	number of control combinations
m_c, m_d, m_p, m_s, m_t	=	masses, kg
n	=	number of variables in $\boldsymbol{x}$
n_c	=	number of constraints
n_k	=	number of knots
n_n	=	number of nodes
n_s	=	number of segments
n_u	=	number of controls
n_y	=	number of states
p	=	cost state
$\boldsymbol{q}$	=	quaternion vector
R	=	specific gas constant
$\mathbb{R}$	=	real space
$\boldsymbol{r}$	=	position vector, km
r	=	radius, m
r_t	=	nozzle throat radius, mm
T	=	temperature, K
t	=	time
$\mathbb{U}$	=	control space
$\boldsymbol{u}$	=	control vector
$\tilde{u}$	=	feasible control value
u^*	=	prespecified control value
V	=	Lyapunov/energy function
V_c, V_d, V_p	=	volumes, m^3
$\boldsymbol{v}$	=	velocity vector
v_t	=	throat velocity, m/s
$\boldsymbol{v}_{1,2}$	=	eigenvectors
$\boldsymbol{w}$	=	eigenvector imaginary part
$\boldsymbol{x}$	=	parameter vector
$\boldsymbol{y}$	=	state vector
$\hat{\boldsymbol{y}}$	=	state estimate vector
$\boldsymbol{z}$	=	output vector
$\hat{\boldsymbol{z}}$	=	output estimate vector
α	=	eigenvalue real part
$\boldsymbol{\beta}$	=	path constraint vector
β	=	scalar weight
γ	=	specific heat
Δ	=	average segment duration
Δt	=	time interval
λ	=	eigenvalue
ρ_t	=	throat gas density, kg/m^3
ϕ	=	endpoint cost
$\boldsymbol{\psi}$	=	point constraint vector
$\boldsymbol{\omega}$	=	angular velocity vector
ω	=	eigenvalue imaginary part

Subscript

f	=	final value
i	=	vector element counter
j	=	node counter
k	=	knot or segment counter
0	=	initial value

Presented at the 19th AAS/AIAA Space Flight Mechanics Meeting, Savannah, GA, 8–12 February 2009; received 27 February 2009; revision received 5 September 2009; accepted for publication 5 September 2009.

*Capt., U.S. Air Force. Ph.D. Candidate, Department of Aerospace Engineering and Engineering Mechanics, 1 University Station, Mail Stop C0600. Student Member AIAA.

†Assistant Professor, Department of Aerospace Engineering and Engineering Mechanics, 1 University Station, Mail Stop C0600. Member AIAA.

I. Introduction

PREVALENT in many engineering fields are systems composed of interdependent continuous and discrete components or Q1

variables. Although processes exist in nature that are accurately modeled with only continuously varying dynamics, it is often the case that some level of decision making occurs in the process. The decision is made by any number of sources, from natural to man-made technologies, but it is clear that the selection is among a discrete number of options. Thus, a hybrid system results, exhibiting continuously varying and discretely chosen components. It is observed in the literature that this often takes on a hierarchical structure, for which continuous or time-driven dynamics exist at the lower levels, and discrete or event-driven dynamics exist at the higher levels [1,2].

The hybrid control problem motivates methods for determining continuous and discrete control variables that affect a hybrid system. The challenge clearly lies in the dual structure of the problem; although strictly continuous and strictly discrete control methods exist, there is room for developing the methods that treat hybrid systems in the sense of stability and optimality. Existing techniques generally extend the theory of either continuous or discrete systems, using (for example) Lyapunov theory [3–5], classical optimal control theory [2,6], or combinatorial techniques [1,7]. Commonly used are mixed-integer nonlinear programming (NLP) algorithms, such as the branch-and-bound method [8], the generalized Benders decomposition [9], and the outer approximation method [10,11]. Although these algorithms may be effective, they are expensive in the sense of computational efficiency due to their combinatorial nature, and increasing the number of discrete variables subsequently increases the size of the problem exponentially.

An earlier study [12] introduced a new method, using a strictly NLP approach, for treating a class of hybrid control problems involving discrete control variables. Although NLP approaches by other authors [13–15] have been suggested, they have been limited in the number of discrete control variables that can be treated simultaneously. In some cases, only a single discrete variable can be considered; in others, the problem size grows exponentially with the number of variables. Alternatively, the finite set control transcription (FSCT) method addresses problems with multiple independent control variables in a unique way, resulting in a linear relation between the quantity of discrete variables and the size of the resulting parameter optimization problem.

The FSCT method is an enhanced collocation method that employs the Hermite–Simpson [16] integration equation to satisfy differential constraints. The implementation for the FSCT method is presented in detail, and its capability is sampled. The objective of the current investigation is to further explore the capability and utility of the method by demonstrating a range of applications. In so doing, the scope of the method is characterized, ideally inspiring additional applications outside those presented here. The applications presented in this investigation are focused on aerospace systems, as these served to motivate the method's development.

The hybrid system under consideration for this investigation is governed by the dynamics

$$\dot{\boldsymbol{y}} = \boldsymbol{f}(t,\ \boldsymbol{y},\ \boldsymbol{u}) \tag{1}$$

where the vector $\boldsymbol{y} \in \mathbb{R}^{n_y}$ represents continuous state variables, and $\boldsymbol{u}$ consists of n_u control elements limited to finite values as

$$u_i \in \mathbb{U}_i = \{\tilde{u}_{i,1},\ \ldots,\ \tilde{u}_{i,m_i}\} \tag{2}$$

The function $\boldsymbol{f}$ describes the continuous variation of the states in terms of time t and the present values for each state and control.

At first glance, this formulation appears to limit the method to a specific class of hybrid systems: all states are presented as continuous and all controls are presented as discrete. Thus, systems with discrete states or continuous controls are apparently excluded. Previously, the implementation carried this limitation. However, the FSCT method can be tailored to include continuous state and control variables within $\boldsymbol{y}$ and, likewise, discrete states and controls in $\boldsymbol{u}$, to allow for a more general treatment of hybrid systems. However, the necessary adjustments are specific to the system under consideration; therefore, that aspect is not discussed here. However, it is observed that many control variables traditionally modeled as continuous may be more accurately described by a combination of continuous dynamic states and discrete controls. This characteristic is demonstrated later in this document. Thus, the formulation of Eq. (1) is not necessarily restrictive. For continuity and clarity, in this study, the term state implies a continuous variable, whereas control implies a discrete one.

This paper is presented as a series of applications designed to demonstrate the utility and versatility of the FSCT method. Whenever possible, results are compared against those produced using alternative hybrid control methods to articulate particular advantages or ways in which multiple methods can be used in tandem.

The first example considers a switched linear system. Between switches, Lyapunov analysis can be employed to identify the control necessary to achieve asymptotic stability. However, although this approach may lead to asymptotic stability within a given segment (defined between switches), the wrong combination of switches can lead to instability. Here, the FSCT method is useful in identifying the optimal switching strategy that achieves the desired control goals and preserves the overall stability of the system. Employing the aforementioned Lyapunov approach can still provide a suitable initial guess to the process. This is one example for which two methods can work together to form a more robust solution strategy. In turn, the results of the FSCT solution may then be used in adjusting the model predictive control (MPC) law to better meet the optimal control goals.

Minimum-time and minimum-acceleration problems, subject to finite set control, are also considered in the analysis of a simple two-dimensional lunar lander problem. A comparison is presented between a MPC strategy and the FSCT approach. The FSCT solution may subsequently be employed in devising a real-time MPC law.

To demonstrate the extension of the FSCT to problems involving variable actuation, a small spacecraft attitude control problem is also examined. The model assumes the vehicle is equipped with small cold-gas thrusters. Initially, the FSCT method is used for attitude tracking when only fixed-thrust amplitude is possible. The follow-on example then considers the more complex case of variable thrust by introducing a detailed model of the variable-thrust nozzle. The thruster provides variable output through a valve core rod that changes the effective nozzle throat area. The parameters that define the actuation of the valve core rod are subject to finite set control. The FSCT method is employed to determine the optimal actuation of the valve core rod that meets the desired tracking goals.

Before proceeding with the individual examples previously described, it is appropriate to introduce some basic background on the FSCT method [12]. Subsequently, the details of each of the examples previously summarized are presented.

II. Finite Set Control Transcription Method Overview

The FSCT is a formulation of the hybrid optimal control problem as a parameter optimization problem that can be solved using a standard NLP algorithm, such as SNOPT [17]. The following overview is intended for the reader possessing a general understanding of direct optimization techniques for continuously varying parameters. Note that, although the method demonstrated here is rooted in direct collocation, alternative formulations exist that capitalize on the structure of indirect or direct shooting methods.

In the most basic sense, the object of a transcription formulation is to convert the optimal control problem formulated as,

minimize

$$\mathcal{J} = \phi(t_0,\ \boldsymbol{y}_0,\ t_f,\ \boldsymbol{y}_f) + \int_{t_0}^{t_f} L(t,\ \boldsymbol{y},\ \boldsymbol{u})\,\mathrm{d}t \tag{3}$$

subject to

$$\dot{\boldsymbol{y}} = \boldsymbol{f}(t,\ \boldsymbol{y},\ \boldsymbol{u}) \tag{4}$$

$$\boldsymbol{0} = \boldsymbol{\psi}_0(t_0,\ \boldsymbol{y}_0) \tag{5}$$

$$\boldsymbol{0} = \boldsymbol{\psi}_f(t_f,\ \boldsymbol{y}_f) \tag{6}$$

$$\mathbf{0} = \boldsymbol{\beta}(t,\ \boldsymbol{y},\ \boldsymbol{u}) \tag{7}$$

into an NLP problem of the form,

$$\min\ F(\boldsymbol{x}) \tag{8}$$

subject to

$$\boldsymbol{c}\,(\boldsymbol{x}) = [\,\boldsymbol{c}_{\dot{y}}^{T}(\boldsymbol{x}) \quad \boldsymbol{c}_{\psi_0}^{T}(\boldsymbol{x}) \quad \boldsymbol{c}_{\psi_f}^{T}(\boldsymbol{x}) \quad \boldsymbol{c}_{\beta}^{T}(\boldsymbol{x})\,]^{T} = \mathbf{0} \tag{9}$$

Ultimately, $\boldsymbol{x}$ must contain the information necessary to express $\boldsymbol{y}(t)$ and $\boldsymbol{u}(t)$ for $t \in [\,t_0 \quad t_f\,]$. In the resulting NLP problem, an initial guess for $\boldsymbol{x}$ is iterated upon until arriving at a feasible and locally optimal set of values. Note that each problem has a cost function to minimize as well as constraints for the dynamics, the initial and final conditions, and any path constraints imposed on the system. In the previous problem definitions, all constraints are presented as equalities; however, extensions certainly exist for inequality constraints as well. The nature of the transcription formulation dictates both the definition of the parameter vector $\boldsymbol{x}$ and the number and forms of the constraint functions in $\boldsymbol{c}(\boldsymbol{x})$ in the resulting parameter optimization problem. The details of how constraint functions are generated is outside the scope of the current development, although this process is articulated in the previous investigation. Let it suffice here to present the definition of $\boldsymbol{x}$ as optimized in the FSCT formulation, knowing that it is possible to devise Eqs. (8) and (9) to ensure that the original optimal control problem is well represented.

Consider the following definition of the parameter vector used for an optimization with the FSCT method:

$$\boldsymbol{x} = [\cdots \quad y_{i,j,k} \quad \cdots \quad \cdots \quad \Delta t_{i,k} \quad \cdots \quad t_0 \quad t_f\,]^{T} \tag{10}$$

The vector $\boldsymbol{x}$ contains parameters that represent states $y_{i,j,k}$ and times $\Delta t_{i,k}$, t_0, and t_f. One of the key features of this parameterization is that control variables are not among the parameters to be optimized. This is unusual: most collocation and direct shooting methods optimize parameters that directly represent control variables. However, in this case, a unique parameterization is necessary, because the controls are discrete variables, whereas the elements of $\boldsymbol{x}$ (by the nature of NLP) are necessarily treated as continuous variables (although perhaps bounded and subject to constraints). Demonstrated presently, a control history is completely defined by the time elements in the parameter vector.

Let the trajectory defined from initial time t_0 to final time t_f be broken up into n_s segments. The interior separation times between segments are termed knots. These represent instances of time when the discrete control variables switch from one feasible value to another. Suppose each control variable is allowed n_k switches between t_0 and t_f. The result is that $n_s = n_u n_k + 1$, and each control is held constant over each segment.

Define n_n as the number of nodes per segment. A node is a point in time at which the values of the state variables are contained within the parameter vector. Specifically, element $y_{i,j,k}$ in Eq. (10) represents the ith state at the jth node of the kth segment. Then, $\boldsymbol{x}$ contains $n_y n_n n_s$ elements pertaining to all of the states at each node. These state values are used directly in the cost and constraint equations (8) and (9).

The elements $\Delta t_{i,k}$ in $\boldsymbol{x}$ indicate the elapsed time between two control switches for a given control variable. Specifically, $\Delta t_{i,k}$ indicates the amount of time that passes between the control switches at the $(k-1)$th and kth knots for the ith control variable.

The values for each u_i are prespecified between each switching point. Thus, $u_{i,k}^{*}$ indicates the prespecified value of the ith control variable before the kth knot. With a discrete number of feasible values, it is possible to set n_k large enough, such that each possible control value is designated as the actual control value for some duration. During the optimization, the values of $\Delta t_{i,k}$ are determined, indicating the amount of time (possibly zero) that each control value is maintained.

The transcription definition is best interpreted with a visualization, such as Fig. 1. In this conceptualization, consider the hybrid control problem with $n_y = 2$ states and $n_u = 2$ controls, where $\mathbb{U}_1 = \{1, 2, 3\}$ and $\mathbb{U}_2 = \{-1, 1\}$. Next, assume the transcription is selected, such that $n_n = 4$ nodes per segment and $n_k = 5$ switching points per control variable. Thus, the number of segments is $n_s = (2)(5) + 1 = 11$ segments.

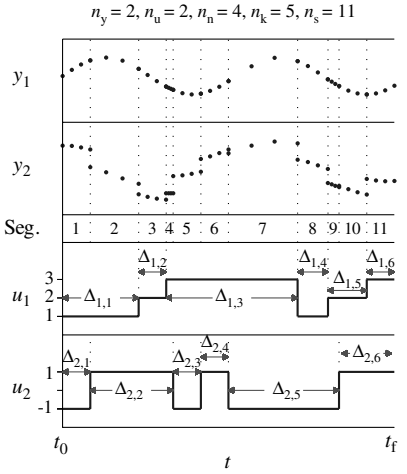


Fig. 1 The parameters of x, where $\Delta_{i,j} = \Delta t_{i,j}$.

It is apparent from Fig. 1 that each control variable may take up to $n_k + 1 = 6$ different values over the trajectory duration. Arbitrarily, the control values are prespecified, so that each control variable systematically switches between the feasible values for that variable. Note that some feasible control values may not be members of the optimal solution. However, through the NLP algorithm, the time durations between switching points are optimized. If one of the prespecified control values is unnecessary or nonoptimal, then the value of the respective time duration is reduced to zero.

Figure 1 further illustrates that the node distribution is not necessarily uniform over the interval $[\,t_0 \quad t_f\,]$. The duration of each segment is dictated by the current values of $\Delta t_{i,k}$. The $n_n = 4$ nodes per segment are evenly distributed over a segment but for shorter segments; this means a closer spacing between nodes. Thus, the state values contained in $\boldsymbol{x}$ may pertain to dense or sparse regions, depending on the time parameters in $\boldsymbol{x}$.

It is also important to note that two nodes are associated with a given knot: the terminal node from the preceding segment and the initial node from the following segment. Therefore, in this parameterization, two sets of state values are contained in $\boldsymbol{x}$ for the times at each knot. For a feasible solution, continuous state variables will have identical values at simultaneous nodes. Constraints in $\boldsymbol{c}(\boldsymbol{x})$ are included to enforce continuity across segments. Of course, these constraints are not always satisfied on intermediate iterations of the solution process. For example, in Fig. 1, the states y_2 are not continuous. Subsequently, this $\boldsymbol{x}$ does not represent a feasible solution. During the FSCT optimization process, elements of $\boldsymbol{x}$ are updated to ensure that, upon completion, the continuity constraints are satisfied.

Additional constraints are included in $\boldsymbol{c}(\boldsymbol{x})$ to ensure that

$$0 = t_f - t_0 - \sum_{k=1}^{n_k+1} \Delta t_{i,k}; \qquad i = 1,\ \ldots,\ n_u \tag{11}$$

Also, at all times, $\Delta t_{i,k} \geq 0$, so that there are no negative time intervals.

By prespecifying the control values, a collocation transcription results for which control switching times are optimized to indicate an optimal control history over all of the feasible control values. Multiple control variables are easily managed and treated completely independently. The control variables for a given segment necessarily affect the hybrid system dynamics, and they are included in appropriate constraint equations for that segment. As the NLP algorithm searches for a feasible and locally optimal set of parameters, the state values are modified at each node so that, upon completion, the state and control histories represent a matching feasible trajectory.

The total number of feasible values for a control variable m_i should have a significant effect on the choice of n_k: the number of switching points allowed over the trajectory. Clearly, when $n_k \gg \max(m_i)$, it is possible to prespecify each control value over several time durations, allowing more flexibility in the resulting NLP problem and a greater likelihood to converge on a small local minimum. However, as n_k gets larger, the sizes of $\boldsymbol{x}$ and $\boldsymbol{c}(\boldsymbol{x})$ also increase, which may complicate (or at least slow down) the optimization process. This characteristic indicates the primary limitation of the FSCT method. To perform an optimization, a user must specify n_k, thus limiting the number of control switches to some maximum value.

In practice, it is useful to overparameterize a problem by setting n_k to an arbitrarily high value, allowing for more control switches than are ultimately necessary. Overparameterizing allows the optimizer to demonstrate the optimal number of switches (less than the parameterized number) by driving to zero the duration of superfluous control values. The overparameterization also allows the user additional flexibility to arbitrarily prespecify control values, knowing that nonoptimal control values are ultimately eliminated in the final solution. Indeed, in the examples that follow, the concept of overparameterization is employed. Consequently, ensuing solutions may display features that are ultimately artifacts of the parameterization. For example, two knots may occur simultaneously, appearing as though the control switches from one value to another and then instantaneously to a third. In the parameterization, zero-duration segments are present, indicating that particular prespecified control values are effectively eliminated from the solution.

One natural characteristic of the numerical optimization process is that it is iterative: a user must supply the optimizer with an initial guess for $\boldsymbol{x}$ and allow the algorithm to improve upon that value until the constraint functions are satisfied and the cost function is minimized. Therefore, the identification of adequate solutions requires a good initial guess $\boldsymbol{x}_0$. In some sense, $\boldsymbol{x}_0$ should be close to the desired solution. The final point $\boldsymbol{x}_f$ will most likely be in the same region as $\boldsymbol{x}_0$. However, it is possible to expand the region containing the initial guess and the final solution by ensuring that the initial guess is not too close to an existing local minimum. The process of balancing these factors makes the determination of a suitable initial guess one of the most difficult aspects of numerical optimization. Thus, it is crucial to provide a philosophy by which effective initial guesses for $\boldsymbol{x}$ can be realized.

In this investigation, $\boldsymbol{x}_0$ is generated by selecting states along or close to a desired (or, maybe, anticipated) solution, whereas controls (that is, time durations) are arbitrarily selected. This approach is convenient, because the analyst often possesses some intuitive knowledge regarding the behavior of the states. For example, initial conditions, final conditions, and path data are often available and can be easily converted into the state values of $\boldsymbol{x}_0$. Once the times for the nodes are known, a guess for $y_{i,j,k}$ can be interpolated. Likewise, starting values of t_0 and t_f can generally be deduced intuitively. Of course, identifying candidate initial values for the control history is not intuitive. In that case, it is convenient to assign initial values for control switches arbitrarily.

This approach can be quite effective. An arbitrarily designated control sequence, combined with an intuitive set of state parameters, generally results in a nonfeasible initial guess. Of course, because the starting point is not feasible, it is also not likely in the vicinity of an existing local minimum. The optimizer adjusts the initial values of all the parameters to make $\boldsymbol{x}$ feasible, moving $\boldsymbol{x}$ away from what may be a poor initial guess. However, by observing that most often $n_y n_n n_s + 2 > n_u(n_k + 1)$ (the number of states is greater than the number of control time parameters), the state elements within $\boldsymbol{x}_0$ provide inertia to keep $\boldsymbol{x}$ in the same vicinity. Thus, this approach helps in producing initial guesses that are not too close or too far away from a good solution.

In this study, the control values are generally prespecified in a manner similar to the profile of Fig. 1. The control values are selected as an ordered set, in which each of the feasible values is present for multiple time durations. One way of expressing the prespecified control values in this example is

$$u^*_{i,k} = \tilde{u}_{i,\mathrm{mod}(k-1,m_i)+1} \tag{12}$$

for $i = 1, \ldots, n_u$ and $k = 1, \ldots, n_k + 1$. This employs the modulus function, where $\mathrm{mod}(a, b) = a - \eta b$, and η is the largest integer multiplier, such that $\eta b \leq a$. This choice can be altered, should problem intuition dictate a less arbitrary arrangement.

An effective strategy for guessing the time durations $\Delta t_{i,k}$ allows for uniform segment durations for all segments in the initial guess. One way of accomplishing this is through the definitions

$$\bar{\Delta} = \frac{t_f - t_0}{n_s} \tag{13}$$

$$\Delta t_{i,k} = \begin{cases} i\bar{\Delta} & k = 1 \\ n_u\bar{\Delta} & k = 2, \ldots, n_k \\ (n_u + 1 - i)\bar{\Delta} & k = n_k + 1 \end{cases} \tag{14}$$

for $i = 1, \ldots, n_u$. This guarantees that, upon the first iteration of the optimization, each segment has duration $\bar{\Delta}$, and the knots are placed as far apart as possible. Also notice that there is a structured rotation between the control variables with regard to switching times: u_1 switches, followed by u_2, and so on. The order of control switches is free to change throughout the optimization process (u_i does not always switch before u_{i+1}), but the initial ordering and initial separation between knot times allows for free movement of the knots in order to arrive at a feasible and locally optimal switching structure.

This philosophy for generating $\boldsymbol{x}_0$ is practiced in each of the applications that follow. In each case, the optimization algorithm successfully converges on a solution, using the initial guess generated as described. However, although this process proves to be effective in these instances, there is no a guarantee that (in general) initial guesses derived using this process will always result in converged solutions. Admittedly, producing a good initial guess is far more of an art than a science (and experience with NLP, in general) and, with the FSCT method specifically, it is beneficial in obtaining and interpreting solutions.

III. Two Stable Linear Systems

Although the FSCT method is especially effective for multiple control variables, first consider a system controlled by only one decision. The system is

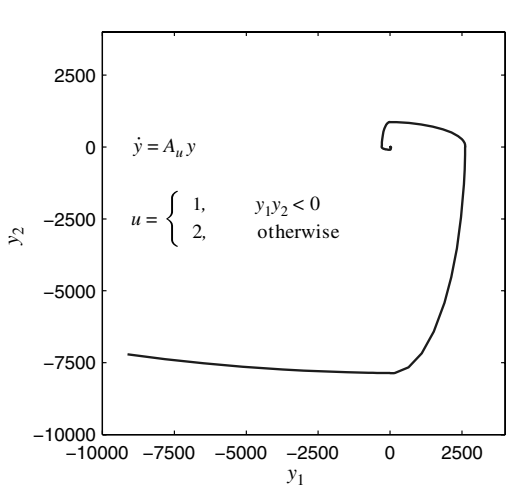
$$\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y}, u) = \boldsymbol{A}_u \boldsymbol{y} \tag{15}$$

$$u \in \{1, 2\} \tag{16}$$

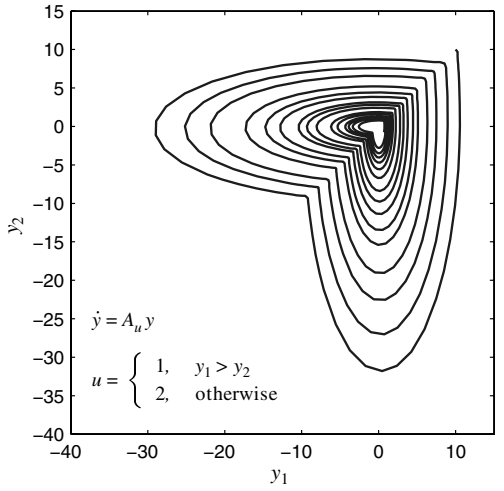
$$\boldsymbol{A}_1 = \begin{bmatrix} -1 & 10 \\ -100 & -1 \end{bmatrix}; \qquad \boldsymbol{A}_2 = \begin{bmatrix} -1 & 100 \\ -10 & -1 \end{bmatrix} \tag{17}$$

Thus, the system is characterized by two separate dynamical modes, and the decision variable determines which of the two is in play at any given time. Notice that, individually, each mode is a linear time-invariant system, guaranteeing exponential stability at the origin $\boldsymbol{y} = \boldsymbol{0}$.

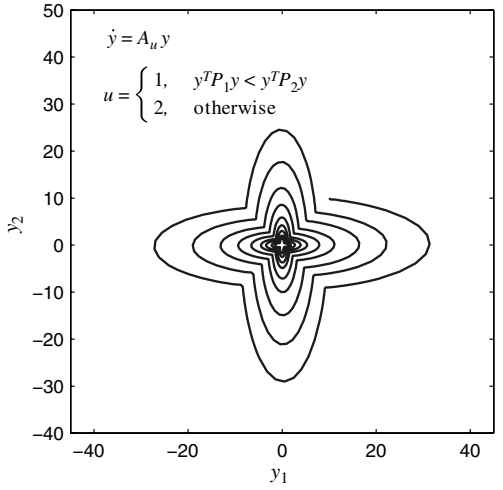
This example is presented by Branicky [3,5] as a classical demonstration of how multiple Lyapunov functions can be used to develop switching laws for the system. It is intriguing in that, although individually stable, one cannot arbitrarily switch between dynamical modes and guarantee system stability. Branicky shows, for example, a switching law devised, such that $u = 1$ when $\boldsymbol{y}$ is in quadrants 2 and 4, and $u = 2$ when $\boldsymbol{y}$ is in quadrants 1 and 3. From any point, the trajectory goes to infinity, as illustrated in Fig. 2a. However, this is not the case for all switching functions. For example, the law that switches modes when $\boldsymbol{y}$ crosses the line $y_2 = y_1$ results in a stable system converging on the origin (Fig. 2b).



a) Control switches when $y_1 y_2 < 0$



b) Control switches when $y_1 > y_2$



c) Control switches when $y^T P_1 y < y^T P_2 y$

Fig. 2 Three switching laws.

The characteristics of the system can be explained via Lyapunov analysis, which follows. The technique of multiple Lyapunov functions is intuitively applied, because the switched system consists of multiple dynamical modes. Subsequently, the FSCT method is applied to demonstrate an alternative analysis technique for determining stable (and optimal) switching laws. This system, presented in Eqs. (15) and (16), serves as an excellent example, because each method can be exercised in a graceful manner due to the inherent simplicity of the linear system. In addition, this example capitalizes on the familiarity of linear systems and the Lyapunov stability theory to the general reader.

A. Stability via Multiple Lyapunov Functions

The key feature of the switching law of Fig. 2b, that guarantees stability, is that the system remains in both modes for exactly one-half of a revolution between each switch. Recall that the two-state linear system with complex eigenvalues $\lambda_{1,2} = \alpha \pm j\omega$ and corresponding eigenvectors $\boldsymbol{v}_{1,2} = \boldsymbol{a} \pm j\boldsymbol{w}$ has a solution of the form:

$$\boldsymbol{y}(t) = e^{\boldsymbol{A}t}\boldsymbol{y}_0 = e^{\alpha t}[\,\boldsymbol{a} \quad \boldsymbol{w}\,]\begin{bmatrix} \cos\omega t & \sin\omega t \\ -\sin\omega t & \cos\omega t \end{bmatrix}[\,\boldsymbol{a} \quad \boldsymbol{w}\,]^{-1}\boldsymbol{y}_0 \tag{18}$$

Then, one-half revolution later from any point,

$$\begin{aligned} &\boldsymbol{y}\left(t + \frac{\pi}{\omega}\right) \\ &\quad = e^{\alpha(t+\frac{\pi}{\omega})}[\,\boldsymbol{a} \quad \boldsymbol{w}\,]\begin{bmatrix} \cos\omega(t+\frac{\pi}{\omega}) & \sin\omega(t+\frac{\pi}{\omega}) \\ -\sin\omega(t+\frac{\pi}{\omega}) & \cos\omega(t+\frac{\pi}{\omega}) \end{bmatrix}[\,\boldsymbol{a} \quad \boldsymbol{w}\,]^{-1}\boldsymbol{y}_0 \\ &\quad = -e^{\alpha\frac{\pi}{\omega}}\boldsymbol{y}(t) \end{aligned} \tag{19}$$

provided that the system remains in the same mode over that time. Thus, for $\alpha < 0$ (a stable system), the function $V = \boldsymbol{y}^T\boldsymbol{y}$, which represents (in a sense) the energy of the system, is guaranteed to be smaller after one-half of a revolution. Consistent switching at intervals of $\frac{\pi}{\omega}$ ensures an incremental decrease in system energy, resulting in convergence to the origin.

Other stable switching structures may also be obtained with a more classical Lyapunov argument. Considering each stable dynamical mode $\boldsymbol{A}_u$ separately, there exist symmetric positive definite matrix pairs $\boldsymbol{P}_u$ and $\boldsymbol{Q}_u$, such that

$$\boldsymbol{P}_u\boldsymbol{A}_u + \boldsymbol{A}_u^T\boldsymbol{P}_u = -\boldsymbol{Q}_u \tag{20}$$

Stability for the mode is demonstrated through the Lyapunov function

$$V_u = \boldsymbol{y}^T\boldsymbol{P}_u\boldsymbol{y} > 0 \tag{21}$$

with the negative time derivative

$$\dot{V}_u = \boldsymbol{y}^T \boldsymbol{P}_u \dot{\boldsymbol{y}} + \dot{\boldsymbol{y}}^T \boldsymbol{P}_u \boldsymbol{y} = -\boldsymbol{y}^T \boldsymbol{Q}_u \boldsymbol{y} < 0 \tag{22}$$

This standard analysis method offers a way of defining a stable switching law according to the behavior of the Lyapunov functions for each mode. For example, define $\boldsymbol{Q}_1 = \boldsymbol{Q}_2 = \boldsymbol{I}$ for simplicity. Then, the Lyapunov equation (20) can be solved uniquely to yield $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$, corresponding to their respective modes. In this case, it is observed that, regardless of the current mode, the energy of the system decreases according to $-\boldsymbol{y}^T \boldsymbol{Q}_u \boldsymbol{y} = -\boldsymbol{y}^T \boldsymbol{y}$. However, $V_1 \neq V_2$, and a reasonable switching law can be selected, such that the Lyapunov function is minimized [4]. Thus,

$$u = \begin{cases} 1 & V_1 \leq V_2 \\ 2 & V_1 > V_2 \end{cases} \tag{23}$$

A trajectory implementing this switching law is illustrated in Fig. 2c.

B. Optimal Switching via Finite Set Control Transcription Method

The method of multiple Lyapunov functions demonstrated previously can be effective in determining switching strategies between a finite number of system modes identified through a single decision variable. Variations on the theme arise by choosing the minimum $\dot{V}_u$ instead of V_u for some candidate Lyapunov functions, or by minimizing some combination of the two [4]. With an infinite set of stable switching laws, a question remains regarding efficiency and robustness. Although various criteria are available to categorize the effectiveness of a switching structure, a simple criterion is presently selected to demonstrate how the FSCT method can aid in the realization of an appropriate switching law. For this example, consider the objective of minimizing the time needed to move a point from its initial position to the vicinity of the origin. Naturally, the trajectory will never go through the origin, as $e^{\alpha t} > 0$ always. However, by choosing a region near the origin, a terminal condition for the optimal control problem is established. Let the final point be subject to

$$\boldsymbol{y}_f^T \boldsymbol{y}_f = 1 \tag{24}$$

such that the objective is to cross the boundary of the unit circle in minimum time, starting from the initial point, $\boldsymbol{y}_0 = [\,10 \quad 10\,]^T$. The optimal control law indicates when to switch between the dynamical modes of $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ to most efficiently guide the trajectory to the terminal manifold.

The FSCT method is well equipped to solve this optimal control problem. Actually, many of the unique characteristics of the solution method are not exercised by this example, due to the fact that the problem consists of only one decision variable. The total number of segments is exactly the number of prespecified control values and, consequently, the control characteristics of each segment are known a priori. Thus, the optimization process simply determines appropriate switching times between segments.

To begin the process, a user selects the number of knots, indicating the total allowable control switches over the course of the trajectory. Let $n_k = 20$ kt for an initial optimization, and prespecify control values, such that

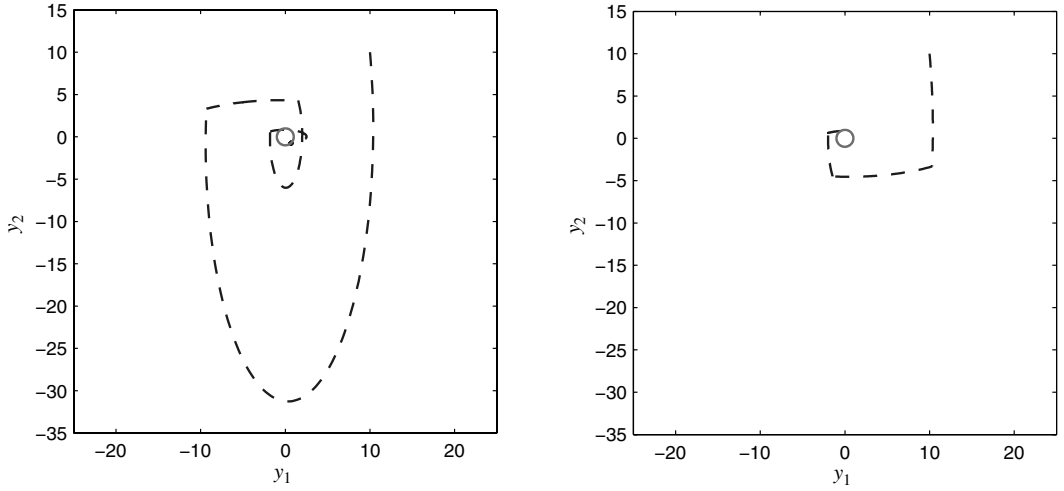
$$u_k^* = \frac{3}{2} + \frac{1}{2}(-1)^k \tag{25}$$

indicating that u begins at the value 1 and alternates between 1 and 2 over each of $n_k + 1 = 21$ segments. Additionally, a user selects a node count that sufficiently captures the state dynamics between control switches when state continuity conditions are satisfied. For this example, $n_n = 100$. Appropriate knot conditions are identified to ensure state continuity across segments, and the optimization function, $\mathcal{J} = F(\boldsymbol{x}) = t_f - t_0$, completes the FSCT formulation.

A preliminary guess is necessary to conduct the nonlinear optimization. The initial point $\boldsymbol{x}_0$ is generated, using an interpolation of the trajectory determined by the minimum Lyapunov function switching law of Fig. 2c over the time interval $t \in [\,0 \quad 3\,]$. Thus, the preliminary cost of the optimization is 3, a reasonable estimate when considering this trajectory first crosses the unit circle at time $t = 3.17$. The knots (switching times) between dynamical modes are uniformly spaced in time from 0 to 3. Thus, the preliminary guess does not satisfy the continuity constraints: the guessed control switches do not correspond to the control switches of the interpolated states. This is acceptable, as the optimization process ensures that the final solution is feasible, as well as locally optimal.

An FSCT optimization applied for the selected initial guess leads to the trajectory illustrated in Fig. 3a. The final time is $t_f = 0.3782$, significantly smaller than the initial guess. The solution is feasible, and three control switches are clearly observable by the corners in the trajectory. With 20 kt, then, it is apparent that 17 possible control switches are not used. Indeed, the solution consists of many knots occurring simultaneously, resulting in zero-duration segments. Thus, the transcription is overparameterized for this solution. Observe that the control switches occur at the following states:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_{k=1} = \begin{bmatrix} -9.3125 \\ 3.3180 \end{bmatrix}; \qquad \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_{k=2} = \begin{bmatrix} 1.5297 \\ 4.2932 \end{bmatrix}$$
$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_{k=3} = \begin{bmatrix} -1.7921 \\ 0.6385 \end{bmatrix} \tag{26}$$



a) Switching law initial guess: $y^T P_1 y < y^T P_2 y$ **b) Switching law initial guess:$-\frac{1}{m} \leq \frac{y_2}{y_1} \leq m$**

Fig. 3 FSCT locally optimal solutions based on various initial control switching strategies.

Notice that the switching points are related, as

$$-\left(\frac{y_1}{y_2}\right)_{k=1} = \left(\frac{y_2}{y_1}\right)_{k=2} = -\left(\frac{y_1}{y_2}\right)_{k=3} \equiv m \tag{27}$$

This ratio implies the switching law

$$u = \begin{cases} 1 & -\frac{1}{m} \le \frac{y_2}{y_1} \le m \\ 2 & \text{otherwise} \end{cases} \tag{28}$$

where $m = 2.8067$. It is important to observe that the trajectory that resulted from the FSCT optimization shows control switches in only quadrants 1 and 2, whereas the control law in Eq. (28) observes switches in each of the four quadrants. The difference is explained through the fact that the solution method, like any NLP method, is only capable of determining locally optimal solutions, most likely in the vicinity of the initial guess. Obviously, the solution is only a local minimum, as the trajectory actually crossed the terminal radius at one point before the final time. In this case, the initial guess and the parameterization leads to a solution with only three observable control switches. However, the symmetry of the trajectories generated in either dynamical mode imply that symmetry should also exist in the switching law. This intuition leads to Eq. (28). To validate this control law, a second optimization problem is solved, this time with a new initial guess. For the second optimization, the states and control switch times of the initial guess are generated using Eq. (28). Optimizing this initial guess, the trajectory of Fig. 3b is determined. Validating the control law, this second solution corresponds perfectly to the initial guess, except that in the final solution, $m = 3.0979$, a slightly larger slope for the switching line. However, the cost is even further improved, with $t_f = 0.0870$.

The fact that the slope value m changes between the two optimizations is not overly surprising. One reason for this is simply that, in each case, the solution is a local, not global, minimum. Through further analysis, it is apparent that m is a factor of the initial point and the size of the terminal radius as well. Indeed, a change to the terminal radius, such that $\boldsymbol{y}_f^T \boldsymbol{y}_f = 0.5$ yields $m = 3.7786$ in the optimal solution.

The intent of this example is to demonstrate how a classical problem, which can be solved using traditional control techniques, can also be analyzed using the FSCT method. One advantage of the latter is the ability to optimize a control solution or control law according to a specified objective. In this case, the final time is minimized; however, it might be equally useful to minimize the integral of the system energy over a fixed time, for example. Both costs capture, in a sense, the sentiment to drive a trajectory to the origin in an efficient manner, although both undoubtedly yield different solutions. It is observed, after all, that the trajectories of Fig. 3 reach the unit circle quickly, but their control law does not guarantee that the trajectory will remain within that circle for all future time (it may escape the region and reenter). Thus, the FSCT method can only guarantee optimality over the range of time considered, not beyond.

IV. Lunar Lander

In a second example, it is useful to revisit the classical lunar lander problem explored in earlier studies [12] and in many sources on optimal control theory [18–20]. The system is modeled in terms of four states and two independently switching control variables subject to finite set control. This, in a sense, increases the complexity of the previous example. Specifically, by implementing multiple control values, the unique segment-switching characteristics of the FSCT method can be observed while maintaining problem simplicity and familiarity. The objective of the problem is to transfer a rocket from a lunar orbit to the lunar surface in minimum time or by using minimum acceleration. The dynamics are constructed in two dimensions, in which separate fixed-magnitude thrusters are pointed in the principal directions. The dynamics are described by

$$\dot{\boldsymbol{y}} = \begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \\ \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ u_1 \\ -g + u_2 \end{bmatrix} \tag{29}$$

where r, v, and u represent position, velocity, and control acceleration, respectively, and the subscripts indicate the horizontal and vertical dimensions. A gravitational constant of $g = 1.6231$ m/s^2 is used. With initial conditions, $\boldsymbol{r}_0 = [200 \quad 15]^T$ km and $\boldsymbol{v}_0 = [-1.7 \quad 0]^T$ km/s, and final conditions, $\boldsymbol{r}_f = \boldsymbol{v}_f = \boldsymbol{0}$, the lander must achieve a soft landing on a specified target from a completely specified initial state. Both minimum-time and minimum-acceleration optimizations are realized with the finite set control constraints, $u_1 \in \{-\tilde{u}_1, 0, \tilde{u}_1\}$ and $u_2 \in \{-\tilde{u}_2, 0, \tilde{u}_2\}$, where $\tilde{u}_1 = 50$ m/s^2 and $\tilde{u}_2 = 20$ m/s^2. The control constraints ensure constant thrust acceleration during thrusting arcs.

A. Optimal Minimum-Time and Minimum-Acceleration Solutions

The FSCT is employed next for the analysis of the lunar lander problem previously described. For this example, let $n_n = 5$ nodes per segment and $n_k = 14$ kt per control axis. In addition, let the prespecified controls be identified as

$$u_{i,k}^* = \tilde{u}_i \cos\left[\frac{\pi}{2}(k-1)\right] \tag{30}$$

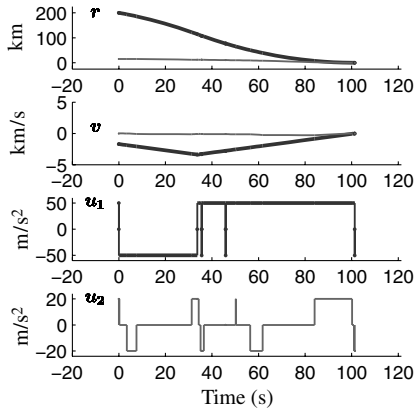
Thus, it is assumed in the control sequence that the vehicle thrusts initially in the positive directions (uprange and up), then it coasts, then it thrusts in the negative directions (downrange and down). The resulting optimizations determine the appropriate times for all control switches, indicating the durations for each thrusting and coasting arc.

An initial guess is devised with $t_0 = 0$ and $t_f = 300$ s, and all knot times are evenly distributed over the interval, such that each segment duration is identical. The state parameters in $\boldsymbol{x}$ are constructed to create a linear progression in each state, from its initial value to its final value. Initial, final, and knot condition constraints are satisfied by the $\boldsymbol{x}$ supplied to the optimizer before the first iteration, but continuity constraints are not immediately satisfied. During the optimization process, $\boldsymbol{x}$ is improved, such that all constraints are satisfied. In addition, the final $\boldsymbol{x}$ minimizes the objective function, representing $\mathcal{J} = t_f - t_0$ for minimum time or

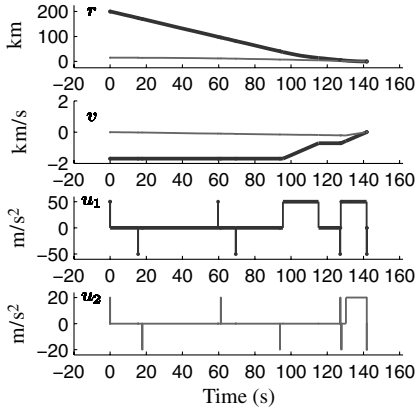
$$\mathcal{J} = \int_{t_0}^{t_f} \boldsymbol{u}^T \boldsymbol{u} \, \mathrm{d}t$$

for minimum acceleration.

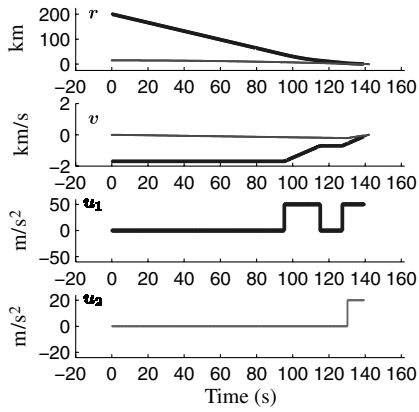
Figure 4 displays the solutions of both the minimum-time and minimum-acceleration problems. Vehicle positions, velocities, and controls are plotted for both minimizations. Notice the control history u_1 for the minimum-time solution. In essence, this solution represents bang–bang control in the first axis, with $u_1(t) = -\tilde{u}_1$ on $t \in [0 \quad 33.66]$ s, and $u_1(t) = \tilde{u}_1$ for the remaining time until t_f, at 101.32 s. Of course, this control behavior is expected for a minimum-time optimization. However, recall that the prespecified initial value for u_1 is $\tilde{u}_1$. As the illustration demonstrates, there is an instantaneous switch in the control at $t_0 = 0$ from $\tilde{u}_1$ to 0 and then from 0 to $-\tilde{u}_1$. The solution exhibits that $\Delta t_{1,1} = \Delta t_{1,2} = 0$ in order to accomplish this. In addition, there are instantaneous switches, approximately located at $t = 34$ and $t = 45$ s. At each of these times, there exist time durations $\Delta t_{1,k}$ for coasting and negative-direction thrusting, and each has been optimized to be identically zero. This behavior is a common artifact of the FSCT formulation. It does not indicate that control switches should occur at these times; rather, it indicates that the problem has been overparameterized with more knots than necessary. However, because control values are prespecified in the optimization, it is useful to overparameterize the problem, allowing for more control switches than needed. Overparameterizing allows the optimizer to demonstrate the optimal number of switches (less than the parameterized number) by driving the superfluous control



a) Minimum time (FSCT)



b) Minimum acceleration (FSCT)



c) Minimum acceleration (MPC)

Fig. 4 Lunar lander example: comparison of FSCT and MPC simulation results.

axis durations to zero. The overparameterization also allows the user additional flexibility to arbitrarily prespecify control values, knowing that nonoptimal control values are eliminated in the final solution.

This same behavior is observed for the minimum-acceleration optimization displayed in Fig. 4b. One may easily observe that most thrusting arcs are reduced exactly to zero by the optimizer for both control axes. This indicates that far fewer switches were necessary to identify this local minimum, and it provides confidence that the formulation has not underparameterized the problem by providing too few control switching opportunities.

For both the minimum-time and minimum-acceleration examples, solutions have been obtained for several values of n_k to assess the effectiveness of the solution method with fewer or more control switches. Numerical evidence suggests that underparameterizing can lead to nonconvergence, but overparameterizing does not affect the final solution, aside from increasing the size of the optimization problem and potentially increasing the convergence time. In these particular examples, varying n_k showed minimal impact on convergence time and no impact on the performance index of the final solution.

An important discovery from the lunar lander example is the extent to which the FSCT method results in implementable control solutions. First, it is clear that the solution requires some interpretation. Superfluous control switches must be discounted before implementing the control history. Actuators with minimum ontimes do not support thrust durations approaching zero; however, within the tolerance of the optimization, zero or near-zero burn durations actually indicate that the respective actuation is not desirable. Clearly, an optimization must be scaled properly in time to differentiate short actuation times from nonoptimal control sequences. Secondly, once a control solution is adequately interpreted, the performance of the solution in a continuous time setting can be nearly identical. Although this collocation technique does rely on a time discretization along each segment, the switching times between control values are optimized over a continuous spectrum. Therefore, the control solution represents exact switching times within the tolerance of the optimization.

B. Model Predictive Controller for Real-Time Implementation

One potential drawback of the FSCT method is that, although capable of producing optimal control histories for the finite set control problem, optimal control laws for real-time implementation are not immediately available. For the general dynamical problem, there is no guarantee that an optimal control solution will imply a real-time law, $\boldsymbol{u} = \boldsymbol{u}(t, \boldsymbol{y})$. To compensate for this limitation, a process is now considered by which FSCT solutions may be implemented in conjunction with a MPC design for real-time implementation of finite control. To begin, a simple model predictive controller is introduced. More complicated, and perhaps more robust, control designs are beyond the scope of this work, although developed in the theory on MPC [21].

1. *Linear Discrete-Time Model*

MPC offers a method for tracking an arbitrary reference trajectory while optimizing a performance index over a finite time horizon. Specifically, MPC techniques can be used for tracking (i.e., implementing) an a priori discovered FSCT solution. A basic model predictive controller is derived using a discrete-time linear dynamic model of the form

$$\boldsymbol{y}(t + \Delta t) = \boldsymbol{A}(t)\boldsymbol{y}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t) \tag{31}$$

$$z(t) = \boldsymbol{C}\boldsymbol{y}(t) \tag{32}$$

Here, $z(t)$ is the measured output from the linear system. In general, then, the nonlinear continuous dynamics of Eq. (1) must be transformed into the form of Eqs. (31) and (32) through appropriate definitions of $\boldsymbol{A}(t)$, $\boldsymbol{B}(t)$, and $\boldsymbol{C}$. It is beyond the scope of this document to demonstrate this transformation.

2. Model Predictive Control Law

The MPC law exploits the linear discrete-time model to develop estimates for the observation variables, $z(t)$, at future time intervals, given the current state values. The output predictions are determined for a finite horizon of future times, and current control values are chosen, so that these output estimates are as close as possible to desired values of the nominal trajectory (the FSCT solution). In the traditional sense, the linear discrete formulation allows controls to be determined by solving a linear equation, holding the control values over the increment Δt constant. The finite set control nature of the hybrid system motivates a minor modification to this design, featured presently.

Let the estimate on the output at time $t + j\Delta t$, given the states at time t, be denoted as $\hat{z}(t + j\Delta t|t)$, such that

$$\hat{z}(t + j\Delta t|t) = \boldsymbol{C}\hat{\boldsymbol{y}}(t + j\Delta t|t) \tag{33}$$

$$\begin{aligned} &= \boldsymbol{C}[\hat{\boldsymbol{A}}(t + (j-1)\Delta t|t)\hat{\boldsymbol{y}}(t + (j-1)\Delta t|t) \\ &\quad + \hat{\boldsymbol{B}}(t + (j-1)\Delta t|t)\boldsymbol{u}(t + (j-1)\Delta t|t)] \end{aligned} \tag{34}$$

In this notation, the symbol $\hat{}$ indicates that the variable is estimated for a future time value. Equation (33) can be manipulated to show that any future estimate may be expressed as a function of $\boldsymbol{y}(t)$, the controls $\boldsymbol{u}(t)$ through $\boldsymbol{u}[t + (j-1)\Delta t]$, and the estimated values of the $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices through time $t + (j-1)\Delta t$. Let the predicted future outputs at discrete-time intervals be stored in the vector $\boldsymbol{Z}$, where

$$\boldsymbol{Z} = \left[\, \hat{z}^T(t + \Delta t|t) \quad \cdots \quad \hat{z}^T(t + j\Delta t|t) \quad \cdots \quad \hat{z}^T(t + p\Delta t|t) \,\right]^T \tag{35}$$

and p is the prediction interval over which observation variables are stored. Using the simplifying assumption that $\boldsymbol{u}(t + j\Delta t) = \boldsymbol{u}(t)$ over the entire prediction interval, the output can be expressed as

$$\boldsymbol{Z} = \boldsymbol{G}\boldsymbol{y}(t) + \boldsymbol{K}\boldsymbol{u}(t) \tag{36}$$

with appropriate definitions for the matrices $\boldsymbol{G}$ and $\boldsymbol{K}$. In Eq. (36), a linear relationship is defined between the current states, the current control values, and the estimates of the future output. Let the nominal output (corresponding to the FSCT trajectory) be expressed at the same discrete-time intervals in the vector $\boldsymbol{Z}_n$. A cost function of the form

$$\mathcal{J} = \frac{1}{2}(\boldsymbol{Z}_n - \boldsymbol{Z})^T\boldsymbol{Q}(\boldsymbol{Z}_n - \boldsymbol{Z}) + \frac{1}{2}(\boldsymbol{u}_n - \boldsymbol{u})^T\boldsymbol{R}(\boldsymbol{u}_n - \boldsymbol{u}) \tag{37}$$

may be employed to penalize deviations in both states and controls away from the nominal. Thus, $\mathcal{J}$ can then be minimized according to user-defined weight matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ to produce a tracking trajectory that also minimizes control over the prediction interval. If $\mathcal{J}$ is minimized at each interval, then $\boldsymbol{u}$ may still take on different values at each interval, even though the prediction equation assumes otherwise. A MPC law for the hybrid system is defined according to

$$\boldsymbol{u} = \arg\min_{\boldsymbol{u}\in\mathbb{U}^{n_u}} \mathcal{J} \tag{38}$$

Simply stated, the implemented control at the current time is the feasible control combination that minimizes the cost function. Thus, if each control variable, u_i, may assume m_i possible values, then the control must consider each of the $\bar{m}$ possible control combinations, implementing the minimizing choice. Notice that

$$\bar{m} = \prod_{i=1}^{n_u} m_i$$

and the control law is most effective for n_u and m_i (and therefore $\bar{m}$), which is reasonably small to reduce the number of computations per interval.

3. Comparison of Model Predictive Control Performance

The hybrid system model predictive controller is easily demonstrated in conjunction with the FSCT method, using the minimum-acceleration solution of the lunar lander problem. With $\bar{m} = (3)(3) = 9$ possible control combinations, it is reasonable to assume that there exists a time interval Δt, such that $\bar{m}$ evaluations of $\mathcal{J}$ can be compared to determine the minimizing control per interval. For this simulation, Δt is chosen, such that there are 500 intervals between t_0 and t_f (less than four intervals per second for the minimum-acceleration trajectory). In addition, a prediction horizon is selected where $p = 10$, indicating that the controller calculates the estimates for the next 10 intervals of the output at each time step. Using Eq. (37), the objective is to mimic the FSCT solution as close as possible via a real-time implementation. Because the a priori discovered optimal solution includes state and control values, it is logical to use all of this information in the controller. The weight matrices, $\boldsymbol{Q}$ and $\boldsymbol{R}$, are proportioned, however, to emphasize position-state tracking over velocity or control tracking.

The results of a simulation implementing the real-time controller are depicted in Fig. 4c, displaying positions, velocities, and control values of the lunar lander. For positions and velocities, both the FSCT solution and the MPC simulation states are plotted to demonstrate minimal deviations between the two. It is especially interesting to compare the control histories of Figs. 4b and 4c: they are nearly identical. The primary observable difference between the MPC simulation and the FSCT solution is that the simulation has removed the instantaneous control switches that resulted from over-parameterization in the FSCT formulation. Thus, with the FSCT solution in hand, it is possible to derive a real-time control law that very closely recreates the optimal trajectory.

The consistency between the FSCT minimum-acceleration solution and the hybrid-system MPC simulation suggests the effectiveness of using the two methodologies in tandem. It is observed that the FSCT method offers control histories instead of implementable control laws. On the other hand, an MPC-derived controller may only be as good as the nominal trajectory selected for tracking. As a pair, however, it is possible to derive optimal trajectories and control histories and implement them in a real-time context, in which perturbations, modeling errors, and other unknowns are likely to arise. This example is intended to further illustrate the utility of the FSCT method when a control law, rather than a control history, is desired.

V. Small Spacecraft Attitude Control

In a final example, the FSCT method is applied to determine finite set control schedules for tracking an arbitrary spacecraft orientation. This example is specifically motivated by spacecraft limited to commercial off-the-shelf actuator technologies that are inexpensive and readily available. The available literature [22–24] indicates a range of new thruster technologies for small spacecraft that are currently under development. Although these may offer wide ranges of thrust magnitudes and performance efficiencies, it is interesting to explore how the capability of traditional technologies can be stretched to maximize performance. The attitude control problem offers an exciting dynamic environment along with conceivable control limitations, which makes the FSCT method quite relevant.

Consider a low-cost microsatellite employing a basic nitrogen cold-gas propulsion system [25,26] for attitude control. Two scenarios are now investigated for this small spacecraft-attitude-control problem. In both of the scenarios, the spacecraft is equipped with six thruster pairs situated on a cuboid spacecraft body to provide purely rotational control in each of the body's principal directions,

positive and negative. The propellant N_2 is stored in a single spherical tank located at the center of mass of the cuboid spacecraft. Temperature and pressure are regulated at each thruster nozzle to allow for constant thrust of 2 N. Pertinent statistics for the spacecraft and the propulsion system are listed in Table 1.

The first scenario demonstrates the simplest control system to conceive. Each thruster pair is controlled by an on–off valve, and thrust magnitudes are limited to two values (2 N when on and 0 N when off). The second scenario explores a variable-thrust cold-gas propulsion system in which the effective throat size of each thruster nozzle varies to alter propellant mass flow. However, the new problem can still be modeled in a finite set control formulation, so that the FSCT method can be used. In transitioning between the two scenario formulations, it is suggested that many variable control problems are actually, at some level, finite control problems with an extended dynamic description.

For the dynamical relations that follow, it is necessary to identify the principal moments of inertia for the spacecraft. Assume a constant mass density within the propellant tank, and further assume a constant mass density in the remaining dry space of volume V_d. Using the quantities derived in Table 1, the spacecraft moment in the first principal direction is

$$J_1 = \frac{1}{12} m_c (l_2^2 + l_3^2) + \frac{2}{5} m_s r^2 \tag{39}$$

with similar definitions for the second and third.

In all subsequent examples, quaternions are selected to represent the attitude of the vehicle. Three fundamental reference frames are defined. The inertial reference frame i is characterized by unit vectors $\hat{i}_j$ for $j = 1, 2$ and 3. Similarly, a body-fixed frame b is defined and associated with unit vectors $\hat{b}_j$. The specified reference attitude history, then, defines frame r and its unit vectors $\hat{r}_j$.

Let ${}^b\boldsymbol{q}^i = {}^b\boldsymbol{q}^i(t)$ denote the four-dimensional time-varying quaternion that describes the orientation of the spacecraft body frame b with respect to the inertial reference frame i. This four-dimensional quaternion can be decomposed [27] into a three-dimensional vector ${}^b\boldsymbol{q}_v^i$ and a scalar element ${}^bq_0^i$, such that

Q3 Table 1 Relevant quantities for a microsatellite

Parameters	Variables	Values
	Assumed quantities	
Dimensions		
Height of cuboid spacecraft	l_1	1.00 m
Length of cuboid spacecraft	l_2	1.25 m
Width of cuboid spacecraft	l_3	1.50 m
Radius of spherical tank	r	0.25 m
Masses		
Dry mass	m_d	15.00 kg
Propellant mass (at t_0)	m_p	5.00 kg
Cold-gas propulsion		
Specific gas constant, N_2	R	296.80 N · m/(kg · K)
Specific heat, N_2	γ	1.4
Storage temperature	T	298.15 K
Maximum thrust	F_t	2.00 N
Nozzle throat radius	r_t	2.50 mm
	Derived quantities	
Volumes		
Cuboid volume	V_c	$l_1 l_2 l_3$
Propellant volume	V_p	$\frac{4}{3}\pi r^3$
Dry volume	V_d	$V_c - V_p$
Masses		
Total mass	m_t	$m_d + m_p$
Cuboid mass	m_c	$m_d(V_c/V_d)$
Extra sphere mass	m_s	$m_t - m_c$
Cold-gas propulsion		
Characteristic velocity	c^*	434.439 m/s
Exhaust velocity	c	787.042 m/s
Gas density/velocity product at throat	$\rho_t v_t$	129.42 kg/(m^{2}s)

$$ {}^b\boldsymbol{q}^i = \begin{bmatrix} {}^bq_0^i \\ {}^b\boldsymbol{q}_v^i \end{bmatrix} \tag{40}$$

Thus, the 3×3 direction cosine matrix ${}^b\boldsymbol{C}^i$ that transforms any three-dimensional vector from inertial to body-fixed coordinates may be expressed as [27,28]

$$ {}^b\boldsymbol{C}^i = \boldsymbol{C}({}^b\boldsymbol{q}^i) = ({}^bq_0^{i\,2} - {}^b\boldsymbol{q}_v^{i\,T}\,{}^b\boldsymbol{q}_v^i)\boldsymbol{I} + 2\,{}^b\boldsymbol{q}_v^i\,{}^b\boldsymbol{q}_v^{i\,T} - 2\,{}^bq_0^i[{}^b\boldsymbol{q}_v^i\times] \tag{41}$$

where $[{}^b\boldsymbol{q}_v^i\times]$ is the skew-symmetric cross-product matrix that operates on a vector in the same way as a cross product [29]. Equations (40) and (41) combine notational elements from various sources [27–29] to simplify the overall bookkeeping process in the present example. Specifically, the left and right superscripts on ${}^b\boldsymbol{q}^i$, and its elements, indicate a quaternion that corresponds to ${}^b\boldsymbol{C}^i$, the direction cosine matrix that transforms a vector from i-frame coordinates to b-frame coordinates. This left/right superscript convention is adopted throughout this document.

A. Low-Cost Cold-Gas Thrusters: Fixed-Thrust Attitude Control

Consider a microsatellite with on–off actuation for its six attitude control thruster pairs. Each thruster delivers either 0 or 2 N of thrust, depending on the state of each on–off valve. Using the existing propulsion system, the objective in this scenario is to track an arbitrary reference trajectory as well as possible, while minimizing fuel expenditure.

Let the desired spacecraft orientation be characterized by ${}^r\boldsymbol{q}^i = {}^r\boldsymbol{q}^i(t)$, where ${}^r\boldsymbol{\omega}^i = {}^r\boldsymbol{\omega}^i(t)$ represents the associated time-varying angular velocity. Furthermore, let the initial attitude of the vehicle, at $t = t_0$, be defined by ${}^r\boldsymbol{q}^i(t_0) = [\,1 \quad 0 \quad 0 \quad 0\,]^T$ and the reference angular velocity be given by

$$ {}^r\boldsymbol{\omega}^i(t) = \begin{bmatrix} 0.3\cos t(1 - e^{-0.01t^2}) + (0.08\pi + 0.006\sin t)te^{-0.01t^2} \\ 0.3\sin t(1 - e^{-0.01t^2}) + (0.08\pi + 0.006\cos t)te^{-0.01t^2} \\ 1 \end{bmatrix} \text{rad/s} \tag{42}$$

Notice that the reference trajectory is completely specified, indicating the ideal attitude for the spacecraft at all times. The reference angular velocity previously specified is arbitrarily selected for this example. However, the parameters that define this reference motion were specifically selected to yield a challenging nominal path to better test the efficiency of the FSCT method under such circumstances.

Let $m_p = m_p(t)$ denote the propellant mass available. This quantity is time-varying because the propellant mass decreases each time the thrusters are engaged. The state vector, in this case, may be defined as

$$ \boldsymbol{y}(t) = \begin{bmatrix} {}^b\boldsymbol{q}^i(t) \\ {}^b\boldsymbol{\omega}^i(t) \\ m_p(t) \\ {}^r\boldsymbol{q}^i(t) \\ p(t) \end{bmatrix} \tag{43}$$

The scalar state, $p = p(t)$, measures an integral cost for deviations between reference and actual trajectories. The previously stated formulation also allows the reference quaternion, ${}^r\boldsymbol{q}^i$, to be determined at each relevant instant in time by the FSCT method.

The control variables, elements of the vector $\boldsymbol{u}$, indicate the position of the on–off valve for each of the 12 thrusters. Because thrusters, at a minimum, are assumed to act in pairs, it is logical to allow each control variable to indicate the valve position for at least two thrusters. However, it is also observed that each thruster pair has a corresponding thruster pair that acts in an opposing fashion, such that their effects are cancelled when both pairs are on. Thus, consider the control vector $\boldsymbol{u} \in \mathbb{U}^3$, where $\mathbb{U} = \{-1,\ 0,\ 1\}$. Thus, $n_u = 3$, and

each control variable is limited to three values, indicating for each principal axis whether the positive-thrusting pair, the negative-thrusting pair, or neither is in the on position.

The state dynamics for the system are described by the following relations,

$$\dot{\boldsymbol{y}} = \begin{bmatrix} {}^b\dot{\boldsymbol{q}}^i \\ {}^b\dot{\boldsymbol{\omega}}^i \\ \dot{m}_p \\ {}^r\dot{\boldsymbol{q}}^i \\ \dot{p} \end{bmatrix} = \boldsymbol{f}(t,\, \boldsymbol{y},\, \boldsymbol{u}) = \begin{bmatrix} \frac{1}{2}\boldsymbol{E}({}^b\boldsymbol{q}^i){}^b\boldsymbol{\omega}^i \\ -\boldsymbol{J}^{-1\,b}\boldsymbol{\omega}^i \times \boldsymbol{J}^b\boldsymbol{\omega}^i + F_t\boldsymbol{J}^{-1}\boldsymbol{L}\boldsymbol{u} \\ -2\frac{F_t}{c}\sum_{i=1}^{3}|u_i| \\ \frac{1}{2}\boldsymbol{E}({}^r\boldsymbol{q}^i){}^r\boldsymbol{\omega}^i \\ {}^b\boldsymbol{q}^{iT}\boldsymbol{H}({}^r\boldsymbol{q}^i)^T\boldsymbol{H}({}^r\boldsymbol{q}^i){}^b\boldsymbol{q}^i \end{bmatrix} \tag{44}$$

where $\boldsymbol{J} = \mathrm{diag}(J_1,\, J_2,\, J_3)$ is the inertia tensor, $\boldsymbol{L} = \mathrm{diag}(l_1,\, l_2,\, l_3)$ contains the spacecraft dimensions, and (for any given quaternion $\boldsymbol{q}$) the matrices $\boldsymbol{E}(\boldsymbol{q})$ and $\boldsymbol{H}(\boldsymbol{q})$ are given by

$$\boldsymbol{E}(\boldsymbol{q}) = \begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}$$

$$\boldsymbol{H}(\boldsymbol{q}) = \begin{bmatrix} q_1 & -q_0 & -q_3 & q_2 \\ q_2 & q_3 & -q_0 & -q_1 \\ q_3 & -q_2 & q_1 & -q_0 \end{bmatrix} = -\boldsymbol{E}^T(\boldsymbol{q}) \tag{45}$$

A few observations are appropriate in regard to the previous state equations. First, recall that the cost index p represents an integral of the deviations with respect to the reference attitude. This quantity, then, depends explicitly on both ${}^r\boldsymbol{q}^i$ and ${}^b\boldsymbol{q}^i$. Whereas ${}^r\boldsymbol{\omega}^i$ is a prespecified function of time, there is no simple closed form expression that yields the value of ${}^r\boldsymbol{q}^i$ at each point in time. This vector must be determined through numerical integration, which motivates its inclusion in the state vector $\boldsymbol{y}$. Computational overhead is further reduced by defining ${}^b\boldsymbol{q}^i$ and ${}^b\boldsymbol{\omega}^i$ in terms of body-fixed coordinates b. This leads to a diagonal inertia tensor that is easily invertible. Finally, note that the propellant mass flow rate $\dot{m}_p$ is only nonzero when one or more thruster pairs is on.

The cost dynamics $\dot{p}$ are used for evaluating an integral cost. Here, it is desired to minimize deviations between the actual and reference coordinate frames. Consider the following relations:

$$\boldsymbol{C}\,({}^r\boldsymbol{q}^b) = \boldsymbol{C}({}^r\boldsymbol{q}^i)\boldsymbol{C}({}^b\boldsymbol{q}^i)^T \tag{46}$$

$${}^r\boldsymbol{q}_v^b = \boldsymbol{H}({}^r\boldsymbol{q}^i){}^b\boldsymbol{q}^i \tag{47}$$

This implies that if $\hat{\boldsymbol{b}}_j \| \hat{\boldsymbol{r}}_j$, then ${}^r\boldsymbol{q}_v^b = 0$. Thus, a cost function that penalizes ${}^r\boldsymbol{q}_v^{b^T}{}^r\boldsymbol{q}_v^b > 0$ ensures minimal deviations between body and reference coordinate frames. This is equivalent to minimizing the terminal cost $p(t_f)$, subject to an initial cost of $p(t_0) = 0$, because

$$p_f - p_0 = \int_{t_0}^{t_f} \dot{p}\ \mathrm{d}t = \int_{t_0}^{t_f} {}^r\boldsymbol{q}_v^{b^T}{}^r\boldsymbol{q}_v^b\ \mathrm{d}t \tag{48}$$

The complete cost function weighs penalties on trajectory tracking deviations with the amount of propellant mass expelled in tracking the reference. Minimizing the total cost function,

$$\mathcal{J} = \beta_1 p_f - \beta_2 m_{p_f} \tag{49}$$

is equivalent to minimizing tracking deviations and maximizing the final propellant mass when $\beta_1 > 0$ and $\beta_2 > 0$.

The problem is completely defined by identifying the remaining initial states for the optimization. Let the spacecraft begin along the reference trajectory. In this case, ${}^b\boldsymbol{q}^i(t_0) = [1 \quad 0 \quad 0 \quad 0]^T$ and ${}^b\boldsymbol{\omega}_0^i = {}^b\boldsymbol{\omega}^i(t_0)$. In addition, assume the initial propellant mass is $m_p(t_0) = 5$ kg. These assumptions imply there is sufficient propellant available to achieve reasonable trajectory tracking for the interval from $t_0 = 0$ to $t_f = 20$ s.

The fixed-time-optimal control problem detailed previously is solved using the FSCT method to yield a feasible and locally optimal trajectory and control switching schedule. For this sample solution, the selected transcription parameters are $n_n = 5$ nodes per segment and $n_k = 20$ kt, allowing 20 control switches in each u_i over the time interval from t_0 to t_f. The prespecified control values are selected based on the following law:

$$u_{i,k}^* = \cos\left[\frac{\pi}{2}(k-1)\right] \tag{50}$$

This control law alternates between positive, zero, and negative torques for each control variable. Clearly, the control sequence selection resembles that of the lunar lander problem, as this seems to allow substantial flexibility to solve the underlying NLP problem.

The FSCT solution is depicted in Fig. 5, when the cost function is set with equal penalty weights, $\beta_1 = \beta_2$. In Fig. 5a, the trajectory position (quaternion) histories for ${}^b\boldsymbol{q}^i$ and ${}^r\boldsymbol{q}^i$ are shown as the actual and desired trajectories, respectively. This illustration gives a visual sense of how well the trajectory can be tracked, given finite value control limitations. In Fig. 5b, the resulting control history (switching schedule) is depicted. For each control variable, note that the durations of arcs associated with both $u_i = 1$ and $u_i = -1$ are reduced to zero. This indicates that the transcription formulation is not underparameterized.

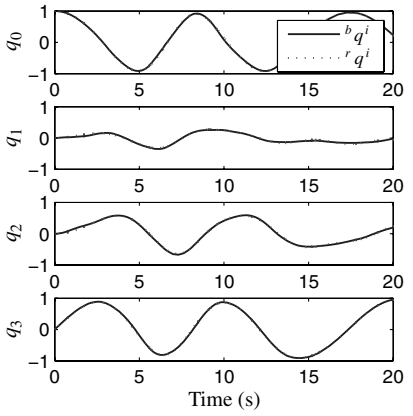
Finally, Fig. 5c depicts the actual and desired angular velocities ${}^b\boldsymbol{\omega}^i$ and ${}^r\boldsymbol{\omega}^i$. It is not unexpected that significantly more deviation is observable in this plot. Control restrictions clearly reduce the way in which velocity variables can change with time. More important, deviations in angular velocities are not penalized in the cost function, so the FSCT method does not attempt directly to drive velocities to match.

B. Low-Cost Cold-Gas Thrusters: Variable-Thrust Attitude Control

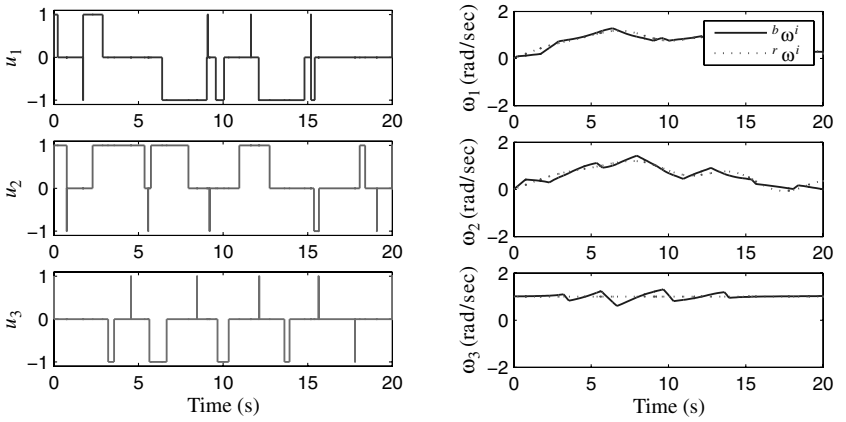
The performance of the previously mentioned system is clearly limited by on–off control actuation. Of course, fixed-thrust control is an obvious choice when the intent is to apply the FSCT method to a real example. Indeed, the simplest (and perhaps least expensive) propulsion systems can benefit from the methodology for determining control strategies for reference tracking. A hybrid system model predictive controller used in conjunction with FSCT solutions may be as successful in this case as it was for the lunar lander presented earlier. Here, however, another scenario is presented to expand the class of applications available to the FSCT methodology.

The most straightforward way of improving upon the solutions of the first attitude control scenario is to expand the solution space to include variable magnitude control inputs. This improves performance through better tracking, less fuel expenditure, or both. Thus, this scenario explores the possibility of a variable amplitude controller with a modified cold-gas propulsion system. The purpose of the development that follows is to demonstrate that the variable control problem can still be interpreted, on a higher level, as a finite set control problem. Extrapolating further, many (if not most) dynamical systems with variable control inputs can be extended to reveal discrete components. Consider, for example, a control system for which the varying inputs are determined by a digital computer. At the highest level, everything is reduced to a binary description (0s and 1s), not unlike the discrete control inputs shown in the examples so far.

A previously developed variable amplitude cold-gas propulsion system [30] serves as the inspiration for the following development. Here, the nitrogen propellant system is modified to allow variation in the effective dimension of the nozzle throat and, subsequently, the propellant mass flow. Consider the illustration of Fig. 6. A valve core rod lies near the throat of the thruster nozzle and has controlled motion up and down. Let the variable d_i indicate the position of the



a) Quaternions: actual vs. desired



b) Control history **c) Angular velocity: actual vs. desired**

Fig. 5 Fixed-thrust attitude control.

valve core rod for the ith thruster. In addition, define r_t as the radius of the nozzle throat. The effective throat area is a function of the rod position, expressed as

$$A_t(d_i) = \pi r_t^2 - \pi\left(r_t - \frac{1}{2}d_i\right)^2 \tag{51}$$

where $0 \le d_i \le 2r_t$. Note that if the rod position is such that $d_i > 2r_t$, no effect is expected on thruster performance, and $(A_t)_{\max} = \pi r_t^2$. Because the throat area directly affects the mass flow through the nozzle (assuming constant propellant density and velocity), it has a direct effect on the magnitude of thrust. Assuming, as before, that the maximum thrust available is $(F_t)_{\max} = 2$ N, then

$$\rho_t v_t = \frac{(F_t)_{\max}}{(A_t)_{\max} c} \tag{52}$$

which can be evaluated using the constants in Table 1. Now, the amplitude of control for each thruster is a function of one discrete variable, indicating the position of the on–off valve and one continuous variable, indicating the valve core rod position. To describe the dynamical system, it is necessary to understand how the rod position d_i is controlled. Surely, there are many ways of doing this, all affecting the nature of the dynamics. Assume then, for the sake of this argument, that each rod is driven by a constant-acceleration motor. Thus, the rod position and its velocity v_i are continuous variables, whereas its acceleration a_i may take only a discrete number of values.

If the valve core rod positions and velocities are included as state variables, a hybrid system ensues consistent with the formulation in Eq. (1), with only continuous states and discrete controls. Although this is not the only formulation for the variable amplitude control problem, this formulation demonstrates that it is possible to extend a variable amplitude control device into a combination of continuous states and discrete decision variables. In this case, states are defined for the physical elements that allow for thrust amplitude variations.

The state vector for this scenario, then, includes the same quantities as the previous scenario, now adding core rod positions and velocities to the set. Recall that, at a minimum, the 12 thrusters of the microsatellite are combined into six thruster pairs. In the first scenario, two pairs providing torque along the same axis of rotation were considered together. In this scenario, the dynamic relations

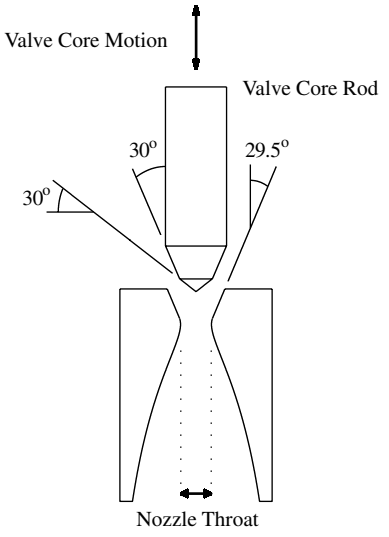


Fig. 6 Variable amplitude thruster nozzle.

dictate that only thruster pairs can be considered to act in harmony. Let the vectors $\boldsymbol{d}$ and $\boldsymbol{v}$, which contain the individual rod dynamics, have six components each. Thus, a thruster pair shares the same rod positions and velocities to ensure that translational accelerations cancel at all times. The state vector for the dynamical system takes the form:

$$\boldsymbol{y} = \begin{bmatrix} {}^{b}\boldsymbol{q}^{i} \\ {}^{b}\boldsymbol{\omega}^{i} \\ m_p \\ \boldsymbol{d} \\ \boldsymbol{v} \\ {}^{r}\boldsymbol{q}^{i} \\ p \end{bmatrix} \tag{53}$$

The control vector is now

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{a} \end{bmatrix} \tag{54}$$

where $\boldsymbol{w}$ and $\boldsymbol{a}$ are each vectors composed of six elements (corresponding to the number of thruster pairs), $w_i \in \{0, 1\}$ indicates whether the ith thruster pair is on or off, and $a_i \in \{-1, 0, 1\}$ indicates the acceleration of the valve core rods of the ith thruster pair, which can be negative, zero, or positive. The dynamics of the system are described by

$$\dot{\boldsymbol{y}} = \begin{bmatrix} {}^{b}\dot{\boldsymbol{q}}^{i} \\ {}^{b}\dot{\boldsymbol{\omega}}^{i} \\ \dot{m}_p \\ \dot{\boldsymbol{d}} \\ \dot{\boldsymbol{v}} \\ {}^{r}\dot{\boldsymbol{q}}^{i} \\ \dot{p} \end{bmatrix} = \boldsymbol{f}(t, \boldsymbol{y}, \boldsymbol{u}) = \begin{bmatrix} \frac{1}{2}\boldsymbol{E}({}^{b}\boldsymbol{q}^{i})^{b}\boldsymbol{\omega}^{i} \\ -\boldsymbol{J}^{-1\,b}\boldsymbol{\omega}^{i} \times \boldsymbol{J}^{b}\boldsymbol{\omega}^{i} + \rho_t v_t c \boldsymbol{J}^{-1}\boldsymbol{L}\boldsymbol{A}(\boldsymbol{d})\boldsymbol{w} \\ -2\rho_t v_t \tilde{\boldsymbol{A}}^{T}(\boldsymbol{d})\boldsymbol{w} \\ \alpha_2 \boldsymbol{v} \\ \alpha_3 \boldsymbol{a} \\ \frac{1}{2}\boldsymbol{E}({}^{r}\boldsymbol{q}^{i})^{r}\boldsymbol{\omega}^{i} \\ {}^{b}\boldsymbol{q}^{iT}\boldsymbol{H}({}^{r}\boldsymbol{q}^{i})^{T}\boldsymbol{H}({}^{r}\boldsymbol{q}^{i})^{b}\boldsymbol{q}^{i} \end{bmatrix} \tag{55}$$

where the previously defined quantities $\boldsymbol{J}$, $\boldsymbol{L}$, $\boldsymbol{E}$, and $\boldsymbol{H}$ are unchanged, and

$$\boldsymbol{A}(\boldsymbol{d}) = \begin{bmatrix} A_t(d_1) & -A_t(d_2) & 0 & 0 & 0 & 0 \\ 0 & 0 & A_t(d_3) & -A_t(d_4) & 0 & 0 \\ 0 & 0 & 0 & 0 & A_t(d_5) & -A_t(d_6) \end{bmatrix} \tag{56}$$

$$\tilde{\boldsymbol{A}}(\boldsymbol{d}) = [A_t(d_1) \quad A_t(d_2) \quad A_t(d_3) \quad A_t(d_4) \quad A_t(d_5) \quad A_t(d_6)]^{T} \tag{57}$$

$$A_t(d_i) = \frac{1}{\alpha_1^2}\left[\pi r_t^2 - \pi\left(r_t - \frac{1}{2}d_i\right)^2\right] \tag{58}$$

Notice immediately that Eq. (58) differs from Eq. (51) by the scaling factor α_1. Additional scaling factors, α_2 and α_3, are present in the valve core rod dynamics as well, so that all state variables remain $\mathcal{O}(10^0)$ to improve the convergence of the underlying NLP problem. In this case, $\alpha_1 = 10^3$ so that r_t and d_i are presented in millimeters. Likewise, $\alpha_2 = 10^1$, and so $\boldsymbol{v}$ is in 10^{-4} m/s, and $\alpha_3 = 10^0$, so that $a_i = 1$ indicates that the ith rod is accelerating by 10^{-4} m/s^2.

Clearly, $\boldsymbol{A}(\boldsymbol{d})$ and $\tilde{\boldsymbol{A}}(\boldsymbol{d})$ represent the effective throat cross-sectional area for each thruster pair, listed in matrix form and vector form, respectively. These facilitate the new definitions for ${}^{b}\dot{\boldsymbol{\omega}}^{i}$ and $\dot{m}_p$. Thus, the effective control torque, evaluated by $\rho_t v_t c \boldsymbol{J}^{-1}\boldsymbol{L}\boldsymbol{A}(\boldsymbol{d})\boldsymbol{w}$ and measured in rad/s^2, as well as the total mass flow defined by $\dot{m}_p$, are determined by the current throat area and the state of the on–off valve.

In as many ways as possible, the optimization of the variable-thrust attitude control scenario is set up identically to the fixed-thrust scenario. The cost function is that of Eq. (49). Again, $\beta_1 = \beta_2$ to allow for a direct comparison between results. In this transcription formulation, $n_n = 5$ and $n_k = 20$ again but, with additional control variables ($n_u = 12$ instead of 3), the total number of segments, and thereby nodes, is significantly increased.

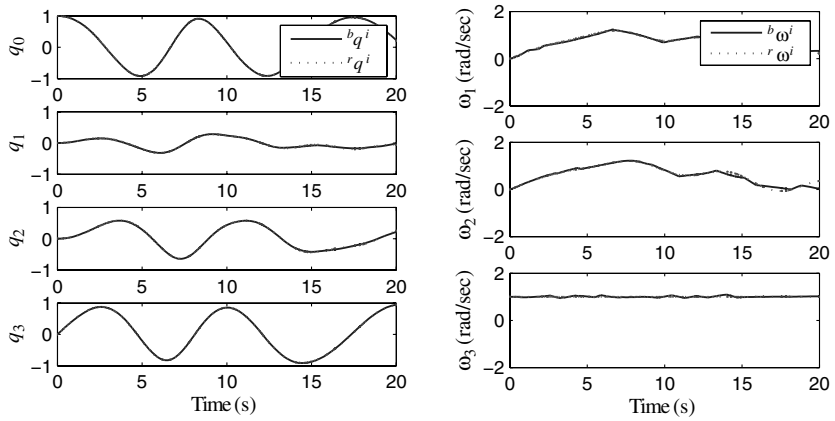
The prespecified controls for the formulation follow a standard structure:

$$w_{i,k}^{*} = \frac{1}{2} + \frac{1}{2}(-1)^{k-1} \tag{59}$$

$$a_{i,k}^{*} = \cos\left[\frac{\pi}{2}(k-1)\right] \tag{60}$$

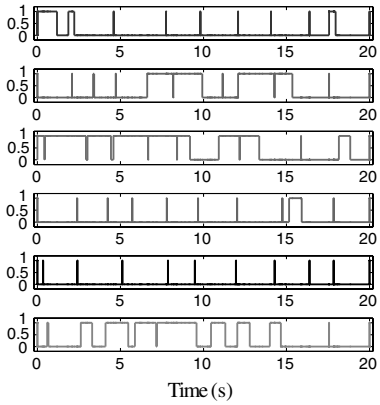
Notice that each w_i alternates between the values 1 and 0, whereas a_i alternates between 1, 0, and -1.

The results of the FSCT optimization are presented in full in Figs. 7–9. Immediately, Figs. 7a and 7b can be compared with Fig. 5 to show how quaternions and angular velocities match the reference trajectories in each scenario. As expected, the variable-thrust

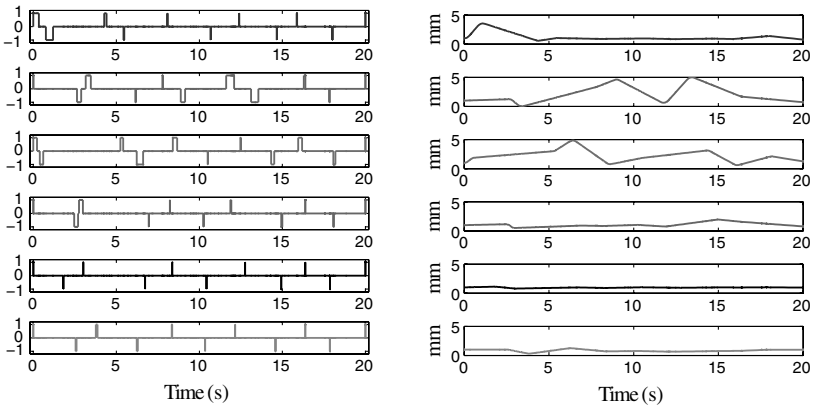


a) Quaternions **b) Angular velocities**

Fig. 7 Variable-thrust attitude control.



a) Thruster pair on-off switch



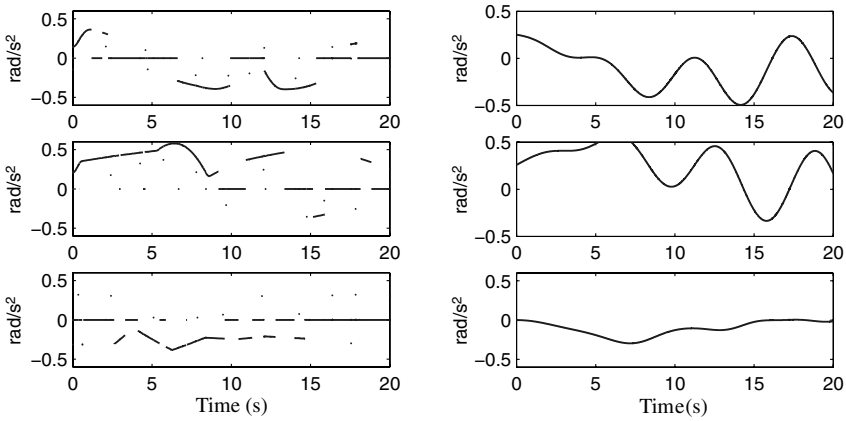
b) Valve accelerator **c) Valve position**

Fig. 8 Variable-thrust attitude control.

formulation offers more flexibility and, consequently, better tracking for the attitude control problem.

Figures 8a and 8b record the control histories that produce this trajectory. For each thruster pair, the controls indicate whether the thruster switch is on or off and whether the motors driving the valve core rod are accelerating the rod. For completeness, Fig. 8c shows the position history of the valve core rods for each thruster pair. Notice that the positions remain within the bounds $0 \leq d_i \leq 5$ mm, where the rod position has an effect on the resulting mass flow through the nozzle.

Figure 9 examines the effective control torque history for the system. When the effects of all of the finite value control variables are considered along with new dynamic states ($\boldsymbol{d}$ and $\boldsymbol{v}$), one can extract the actual control torque, measured in rad/s^2, that is applied to the spacecraft at any time. Figure 9a illustrates this. Note that the zero-duration dots contained in the figure are artifacts of zero-duration segments that naturally result in an FSCT solution. For the sake of this discussion, they can effectively be ignored. As a comparison with this solution, Fig. 9b illustrates the control torque for an unconstrained control system that tracks the reference trajectory perfectly. The unconstrained control torque is derived using a continuous Lyapunov-based control law, which guarantees perfect tracking of quaternions and angular velocities (because initial states are along the reference). Some distinct similarities are easy to



a) FSCT control torque **b) Unconstrained control torque**

Fig. 9 FSCT variable-thrust attitude control torque vs unconstrained attitude control torque.

observe by looking at the plots together. Certainly, control torque magnitudes are similar, but there are also points at which the derived control torque from the finite set formulation very closely mimics the behavior of the purely continuous control.

This is viewed as a significant result that demonstrates how a detailed hybrid system formulation can approach a continuous formulation. Although it is a step backward to use a finite control formulation if control inputs are truly continuous, perhaps it is reasonable to argue that many systems (if not most) are truly hybrid systems, modeled as continuous. Often, it is easier to model the continuous system, as numerous methodologies exist for treating such systems. However, if a system has discrete components, it is ideal to treat them as such. Thus, the FSCT method offers an avenue for modeling any discrete components present in the problem, at whatever level they may appear in the dynamical model.

VI. Conclusions

The FSCT method is demonstrated for the determination of optimal solutions to hybrid control problems. The intent of the present investigation is to explore the range of applications that the FSCT method is suited to treat. Basic and complex applications illustrate its capability and utility.

The FSCT method uniquely transforms the hybrid control problem into a NLP problem to optimize controls limited to finite sets of values. A primary feature demonstrated is the ability to manage multiple independent decision inputs simultaneously. The number of control variables contributes linearly to the parameter size of the underlying NLP problem, as opposed to the exponential growth characterizing similar methodologies. This distinction allows the FSCT method to efficiently optimize systems with many control variables. In this investigation, an application with two control variables articulates the characteristics of a multiple control solution, whereas an application using up to 12 control variables displays the method's effectiveness with large dimensionality. Aside from the computational advantages offered by the FSCT method, this study further demonstrates the utility of the method when used in conjunction with Lyapunov techniques or MPC methods for real-time applications.

Acknowledgment

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the U.S. Air Force, the Department of Defense, or the U.S. Government.

References

[1] 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, Jan. 1998, pp. 31–45. doi:10.1109/9.654885

[2] Gokbayrak, K., and Cassandras, C. G., "Hybrid Controllers for Hierarchically Decomposed Systems," *Hybrid Systems: Computation and Control*, Vol. 1790, Springer–Verlag, Berlin, March 2000, pp. 117–129.

[3] Branicky, M. S., "Stability of Switched and Hybrid Systems," *Proceedings of the 33rd IEEE Conference on Decision and Control*, Vol. 4, IEEE Publ., Piscataway, NJ, Dec. 1994, pp. 3498–3503.

[4] Pettersson, S., and Lennartson, B., "Controller Design of Hybrid Systems," *Hybrid and Real-Time Systems*, Vol. 1201, Springer–Verlag, Berlin, March 1997, pp. 240–254.

[5] Branicky, M. S., "Multiple Lyapunov Functions and Other Analysis Tools for Switched and Hybrid Systems," *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, April 1998, pp. 475–482. doi:10.1109/9.664150

[6] Azhmyakov, V., Attia, S. A., Gromov, D., and Raisch, J., "Necessary Optimality Conditions for a Class of Hybrid Optimal Control Problems," *Hybrid Systems: Computation and Control*, Vol. 4416, Springer–Verlag, Berlin, April 2007, pp. 637–640.

[7] Ball, J. A., Chudoung, J., and Day, M. V., "Robust Optimal Switching Control for Nonlinear Systems," *SIAM Journal on Control and Optimization*, Vol. 41, No. 3, 2002, pp. 900–931. doi:10.1137/S0363012900372611

[8] Floudas, C. A., *Nonlinear and Mixed-Integer Optimization*, Oxford Univ. Press, New York, 1995, pp. 98–107.

[9] Geoffrion, A. M., "Generalized Benders Decomposition," *Journal of Optimization Theory and Applications*, Vol. 10, No. 4, Oct. 1972, pp. 237–260. doi:10.1007/BF00934810

[10] Duran, M. A., and Grossmann, I. E., "An Outer-Approximation Algorithm for a Class Of Mixed-Integer Nonlinear Programs," *Mathematical Programming*, Vol. 36, No. 3, Oct. 1986, pp. 307–339. doi:10.1007/BF02592064

[11] Fletcher, R., and Leyffer, S., "Solving Mixed Integer Nonlinear Programs by Outer Approximation," *Mathematical Programming*, Vol. 66, Nos. 1–3, Aug. 1994, pp. 327–349. doi:10.1007/BF01581153

[12] Stanton, S. A., and Marchand, B. G., "Enhanced Collocation Method for Dynamical Systems Subject to Finite Set Control," *Journal of Guidance, Control, and Dynamics*, 2010 (accepted for publication).

[13] Wei, S., Uthaichana, K., Žefran, M., DeCaralo, R. A., and Bengea, S., "Applications of Numerical Optimal Control to Nonlinear Hybrid Systems," *Nonlinear Analysis: Hybrid Systems*, Vol. 1, No. 2, 2007, pp. 264–279. doi:10.1016/j.nahs.2006.10.007

[14] Gerdts, M., "A Variable Time Transformation Method for Mixed-Integer Optimal Control Problems," *Optimal Control Applications and Methods*, Vol. 27, No. 3, 2006, pp. 169–182. doi:10.1002/oca.778

[15] Sager, S., Bock, H. G., Diehl, M., Reinelt, G., and Schlöder, J. P., "Numerical Methods for Optimal Control with Binary Control Functions Applied to a Lotka–Volterra Type Fishing Problem,"*Recent Advances in Optimization*, Springer–Verlag, Berlin, 2006, pp. 269–289.

[16] Betts, J. T., *Practical Methods for Optimal Control Using Nonlinear Quadratic Programming*, Society of Industrial and Applied Mathematics, Philadelphia, 2001.

[17] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Journal on Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006 doi:10.1137/S1052623499350013.

[18] Kirk, D. E., *Optimal Control Theory: An Introduction*, Prentice–Hall, Englewood Cliffs, NJ, 1970.

[19] Bryson, A. E., and Ho, Y.-C., *Applied Optimal Control: Optimization, Estimation, and Control*, Taylor and Francis, New York, 1975.

[20] Hull, D. G., *Optimal Control Theory for Applications*, Springer–Verlag, New York, 2003.

[21] Sunan, H., Kiong, T. K., and Heng, L. T., *Applied Predictive Control*, Springer–Verlag, London, 2002.

[22] Mueller, J., "Thruster Options for Microspacecraft: A Review and Evaluation of Existing Hardware and Emerging Technologies," 33rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, AIAA Paper 97-3058, July 1997.

[23] Gonzales, D. A., and Baker, R. P., "Microchip Laser Propulsion for Small Satellites," 37th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, AIAA Paper 2001-3789, July 2001.

[24] Phipps, C., Luke, J., and Helgeson, W., "Laser Space Propulsion Overview," *Proceedings of SPIE: The International Society for Optical Engineering*, Vol. 6606, 2007, Paper 660602. doi:10.1117/12.729495

[25] Humble, R. W., Henry, G. N., and Larson, W. J., Eds., *Space Propulsion Analysis and Design*, McGraw–Hill, New York, 1995.

[26] Adler, S., Warshavsky, A., and Peretz, A., "Low-Cost Cold-Gas Reaction Control System for Sloshsat Flevo Small Satellite," *Journal of Spacecraft and Rockets*, Vol. 42, No. 2, Mar–Apr 2005, pp. 345–351. doi:10.2514/1.5128

[27] Hughes, P. C., *Spacecraft Attitude Dynamics*, Wiley, New York, 1994.

[28] Kane, T. R., Likins, P. W., and Levinson, D. A., *Spacecraft Dynamics*, McGraw–Hill, New York, 1983.

[29] Costic, B. T., Dawson, D. M., de Queiroz, M. S., and Kapila, V., "Quaternion-Based Adaptive Attitude Tracking Controller Without Velocity Measurements," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 6, Nov–Dec 2001, pp. 1214–1222. doi:10.2514/2.4837

[30] Stone, W. C., "Fast Variable-Amplitude Cold Gas Thruster," *Journal of Spacecraft and Rockets*, Vol. 32, No. 2, Mar–Apr 1995, pp. 335–343. doi:10.2514/3.26615

D. Spencer
Associate Editor