Real-Time, Propellant-Optimized Spacecraft Motion Planning under Clohessy-Wiltshire-Hill Dynamics

Joseph A. Starek
Aeronautics & Astronautics
Stanford University
Stanford, CA, 94305
650-497-1469
jstarek@stanford.edu

Edward Schmerling, Gabriel D. Maher
Inst. for Computational and Mathematical Engineering
Stanford University
Stanford, CA, 94305
650-497-1469
schmrlng@stanford.edu, gdmaher@stanford.edu

Brent W. Barbee
NASA Goddard Space Flight Center
NASA
Greenbelt, MD, 20771
301-286-1837
brent.w.barbee@nasa.gov

Marco Pavone
Aeronautics & Astronautics
Stanford University
Stanford, CA, 94305
650-723-4432
pavone@stanford.edu

***Abstract*— This paper presents a sampling-based motion planning algorithm for real-time, propellant-optimized autonomous spacecraft trajectory generation in near-circular orbits. Specifically, this paper leverages recent algorithmic advances in the field of robot motion planning to the problem of impulsively-actuated, propellant-optimized rendezvous and proximity operations under the Clohessy-Wiltshire-Hill (CWH) dynamics model. The approach calls upon a modified version of the Fast Marching Tree (FMT*) algorithm to grow a set of feasible and *actively-safe* trajectories over a deterministic, low-dispersion set of sample points covering the free state space. Key features of the proposed algorithm include: (i) theoretical guarantees of trajectory safety and performance, (ii) real-time implementability, and (iii) generality, in the sense that a large class of constraints can be handled directly. As a result, the proposed algorithm offers the potential for widespread application, ranging from on-orbit satellite servicing to orbital debris removal and autonomous inspection missions.**

TABLE OF CONTENTS

1. INTRODUCTION

Real-time autonomy for spacecraft proximity operations near circular orbits is an inherently challenging task, particularly for onboard implementation where computational capabilities are limited. Many effective real-time solutions have been developed for the *unconstrained* case (*e.g.*, state transition matrix manipulation [1], glideslope methods [2], safety ellipses [3, 4], and others [5]). However, the difficulty of real-time processing increases when there is a need to operate near other objects and/or incorporate some notion of propellant-optimality or control-effort minimization. In such cases, care is needed to efficiently handle collision-avoidance, plume impingement, sensor line-of-sight, and other complex guidance constraints, which are often non-convex and may depend on time and a mixture of state and control variables. State-of-the-art techniques for collision-free spacecraft proximity operations (both with and without optimality guarantees) include artificial potential function guidance [6, 7], convexification techniques [8], enforcing line-of-sight or approach corridor constraints [9–11], maintaining relative separation [12], satisfying Keep-Out-Zone (KOZ) constraints using mixed-integer (MI) programming [13], and kinodynamic motion planning algorithms [14–17]. Requiring hard assurances of mission safety with respect to a wide variety and number of potential failure modes [18] provides an additional challenge.

The objective of this paper is to design an automated approach to actively-safe spacecraft trajectory optimization for rendezvous and proximity operations near circular orbits, which we model using Clohessy-Wiltshire-Hill (CWH) dynamics. Our approach is to leverage recent advances from the field of *sampling-based* robot motion planning [19]. Several decades of research have shown that sampling-based planning algorithms (dubbed "planners" throughout this paper) show promise for tightly-constrained, high-dimensional optimal control problems such as the one considered in this paper. Sampling-based motion planning essentially entails the breakdown of a complex trajectory control problem into a series of many relaxed, simpler Two-Point Boundary Value Problems (2PBVPs, or "steering" problems) that are subsequently evaluated *a posteriori* for constraint satisfaction and efficiently strung together into a graph (*i.e.*, a tree or roadmap). In this way, complex constraints like obstacle avoidance or plume impingement are decoupled from the generation of subtrajectories (or graph "edges") between graph states (or "samples"), separating dynamic simulation from constraint checking – a fact we exploit to achieve real-time capability. Critically, this approach avoids the *explicit* construction of the free state space, which is prohibitive in complex planning problems. As a result, sampling-based algorithms can address a large variety of constraints and can provide significant computational benefits with respect to traditional optimal control methods and mixed-integer

978-1-4799-5380-6/15/$31.00

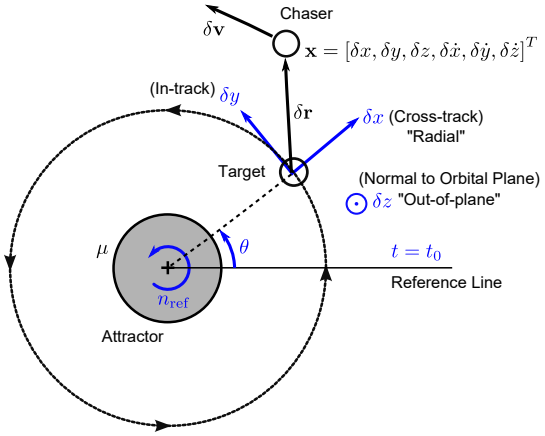
programming [19]. Furthermore, through a property called *asymptotic optimality* (AO), sampling-based algorithms can be designed to provide guarantees of optimality in the limit that the number of samples taken approaches infinity. This makes sampling-based planners a strong choice for the problem of spacecraft control.

Though the aforementioned works [14–17] on sampling-based planning for spacecraft proximity operations have addressed several components of the safety-constrained, optimal CWH autonomous rendezvous problem, few have addressed the aspect of real-time implementability in conjunction with both a 2-norm propellant-cost metric and active trajectory safety with respect to control failures. This paper seeks to fill this gap. The paper's central theme is a rigorous proof of asymptotic optimality for a particular sampling-based planner, namely a modified version of the Fast Marching Tree (FMT*) algorithm [20], under impulsive CWH spacecraft dynamics with hard safety constraints. First, a description of the problem scenario is provided in Section 2, along with a formal definition of the 2-norm cost metric that we employ as a proxy for propellant consumption. Next, we proceed in Section 3 to our proposed approach employing the modified FMT* algorithm. The section begins with presentation of a conservative approximation to the propellant-cost reachability set, which characterizes the set of states that are "nearby" to a given initial state in terms of propellant use. These sets, bounded by unions of ellipsoidal balls, are then used to show that the modified FMT* algorithm maintains its (asymptotic) optimality when applied to CWH dynamics under the 2-norm cost function. From there, in Section 4, the paper presents two techniques for improving motion planning solutions: (i) an analytical technique that can be called both during planning and post-processing to merge Δv-vectors between any pair of concatenated graph edges, and (ii) a continuous trajectory smoothing algorithm, which can reduce the magnitude of Δv-vectors by relaxing the implicit constraint to pass through sample points while still maintaining solution feasibility.

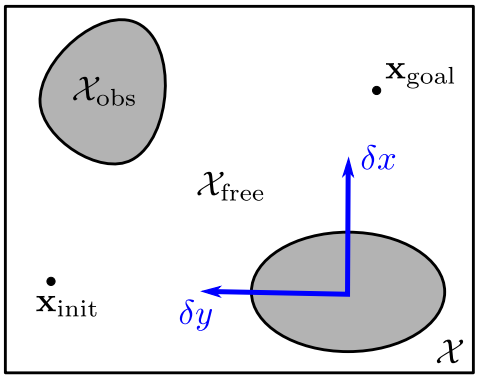
The combination of these tools into a unified framework provides a flexible, general technique for near-circular orbit spacecraft trajectory generation that automatically guarantees bounds on run time and solution quality (propellant cost) while handling a wide variety of (possibly non-convex) state, time, and control constraints. The methodology is demonstrated in Section 5 on a single-chaser, single-target scenario simulating a near-field Low Earth Orbit (LEO) approach with hard constraints on total maneuver duration, relative positioning (including keep-out-zone and antenna interference constraints), thruster plume impingement, individual and net Δv-vector magnitudes, and a two-fault thruster stuck-off failure tolerance. The performances of FMT* and the trajectory smoothing techniques are evaluated as a function of sample count and a propellant cost threshold.

2. Problem Formulation

We begin by defining the problem we wish to solve. We model the near-field homing phase and approach for a spacecraft seeking to maneuver near a target that is moving in a well-defined, circular orbit (see Fig. 1(a)). Let the *state space* $\mathcal{X} \subset \mathbb{R}^d$ represent the d-dimensional region in the target's Local Vertical, Local Horizontal (LVLH) frame in which the mission is defined, and define the *obstacle region* or $\mathcal{X}_{\text{obs}}$ as the set of states within $\mathcal{X}$ that result in mission failure (*e.g.*, states colliding with the target or which lie outside of a specified approach corridor, for example). We then also define



(a) Schematic of CWH dynamics, which models relative guidance near a single target in a circular orbit.



(b) A representative planning query between feasible states $\mathbf{x}_{\text{init}}$ and $\mathbf{x}_{\text{goal}}$.

Figure 1. Illustration of the CWH planning scenario. Here n_{ref} is the mean motion of the target spacecraft orbit, θ is its mean anomaly, t denotes time, $\delta\mathbf{r}$ and $\delta\mathbf{v}$ are the chaser relative position and velocity, and $(\delta x, \delta y, \delta z)$ are the LVLH frame coordinates. The CWH frame rotates with the target at rate n_{ref} as it orbits the gravitational attractor, μ. Planning takes place in the LVLH frame state space $\mathcal{X}$, in which a safe trajectory is sought between states $\mathbf{x}_{\text{init}}$ and $\mathbf{x}_{\text{goal}}$ within the feasible (collision-free) subspace $\mathcal{X}_{\text{free}}$ around nearby obstacles $\mathcal{X}_{\text{obs}}$.

the *free space* or $\mathcal{X}_{\text{free}}$ as the complement of $\mathcal{X}_{\text{obs}}$, *i.e.*, states in $\mathcal{X}$ which lie outside of obstacles. As illustrated in Fig. 1(b), let $\mathbf{x}_{\text{init}}$ represent the chaser spacecraft's initial state relative to the target, and let $\mathbf{x}_{\text{goal}} \in \mathcal{X}_{\text{goal}}$ be a goal state (a new position/velocity near which the chaser can initiate a docking sequence, a survey maneuver, *etc.*) inside the *goal region* $\mathcal{X}_{\text{goal}}$. Finally, define a *state trajectory* (or simply "trajectory") as a piecewise-continuous function of time $\mathbf{x}(t) : \mathbb{R} \to \mathcal{X}$, and let Σ represent the set of all state trajectories. Every state trajectory is implicitly generated by a control trajectory $\mathbf{u}(t) : \mathbb{R} \to \mathcal{U}$, where $\mathcal{U}$ is the set of controls, through the system dynamics $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t)$, where f is the system's state transition function. A state trajectory is called a *feasible* solution to the planning problem $(\mathcal{X}_{\text{free}}, t_{\text{init}}, \mathbf{x}_{\text{init}}, \mathbf{x}_{\text{goal}})$ if: (i) it satisfies the boundary conditions $\mathbf{x}(t_{\text{init}}) = \mathbf{x}_{\text{init}}$ and $\mathbf{x}(t_{\text{final}}) = \mathbf{x}_{\text{goal}}$ for some time $t_{\text{final}} > t_{\text{init}}$, (ii) it is *collision-free*; that is, $\mathbf{x}(\tau) \in \mathcal{X}_{\text{free}}$ for all $\tau \in [t_{\text{init}}, t_{\text{final}}]$, and (iii) it obeys all other trajectory constraints, including the system dynamics. The general motion planning problem can then be defined as follows.

Definition 2.1 (Optimal Planning Problem) Given a planning problem $(\mathcal{X}_{\text{free}}, t_{\text{init}}, \mathbf{x}_{\text{init}}, \mathbf{x}_{\text{goal}})$ and a cost functional J :

$\Sigma \times \mathcal{U} \times \mathbb{R} \to \mathbb{R}_{\geq 0}$, find a feasible trajectory $\mathbf{x}^*(t)$ with associated control trajectory $\mathbf{u}^*(t)$ and time span $t^* = [t_{\text{init}},\, t_{\text{final}}]$ for $t_{\text{final}} \in [t_{\text{init}}, \infty)$ such that $J(\mathbf{x}^*(\cdot), \mathbf{u}^*(\cdot), t^*) = \min\{J(\mathbf{x}(\cdot), \mathbf{u}(\cdot), t) \mid \mathbf{x}(t) \text{ and } \mathbf{u}(t) \text{ are feasible}\}$. If no such trajectory exists, report failure.

For our particular case, we employ a control-effort cost functional J that considers only the control trajectory $\mathbf{u}(t)$ and the final time t_{final}, which we represent by the notation $J(\mathbf{u}(t), t_{\text{final}})$. Tailoring Def. 2.1 to impulsively-actuated propellant-optimal motion planning near circular orbits (where here we assume propellant optimality is measured by the 2-norm metric), the spacecraft motion planning problem we wish to solve is formulated as:

$$
\begin{aligned}
\text{Given:} \quad & \text{Initial state } \mathbf{x}_{\text{init}}(t_{\text{init}}), \text{Goal region } \mathcal{X}_{\text{goal}}, \\
& \text{Free space } \mathcal{X}_{\text{free}} \\
\underset{\mathbf{u}(t), t_{\text{final}}}{\text{minimize}} \quad & J(\mathbf{u}(t), t_{\text{final}}) = \int_{t_{\text{init}}}^{t_{\text{final}}} \|\mathbf{u}(t)\|_2 \,\mathrm{dt} = \sum_{\mathrm{i}=1}^{\mathrm{N}} \|\mathbf{\Delta v}_{\mathrm{i}}\|_2 \\
\text{subject to} \quad & \mathbf{x}(t_{\text{init}}) = \mathbf{x}_{\text{init}} \\
& \mathbf{x}(t_{\text{final}}) \in \mathcal{X}_{\text{goal}} \\
& \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t) \\
& \mathbf{x}(t) \in \mathcal{X}_{\text{free}} \text{ for all } t \in [t_{\text{init}},\, t_{\text{final}}] \\
& \left.\begin{aligned} \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t) \leq 0 \\ \mathbf{h}(\mathbf{x}(t), \mathbf{u}(t), t) = 0 \end{aligned}\right\} \text{for all } t \in [t_{\text{init}},\, t_{\text{final}}] \\
& \exists \text{ safe } \mathbf{x}_{\text{CAM}}(\tau), \tau > t \text{ for all } \mathbf{x}(t)
\end{aligned}
\tag{1}
$$

where t_{init} and t_{final} are the initial and final times, $\mathbf{x}_{\text{CAM}}(\tau)$ refers to an infinite-horizon Collision-Avoidance-Maneuver (CAM), and we restrict our attention to impulsive control laws $\mathbf{u}(t) = \sum_{i=1}^{N} \mathbf{\Delta v}_i \delta(t - \tau_i)$, where $\delta(\cdot)$ denotes the Dirac delta function, which represent a finite sequence of instantaneous translational burns $\mathbf{\Delta v}_i$ fired at discrete times τ_i (note that the number of burns N is not fixed *a priori*). Though one could consider the set of all possible control laws, it is both theoretically and computationally simpler to optimize over the finite-dimensional search space enabled by using Δv-vectors; furthermore, such control laws represent the most common forms of propulsion systems used on-orbit, including high-impulse cold-gas and liquid bi-propellant thrusters, and can at least in theory approximate continuous control trajectories in the limit that $N \to \infty$.

We now elaborate on the objective function and each constraint in turn.

Cost Functional

A critical component of the spacecraft rendezvous problem is the choice of the cost function. Consistent with the conclusions of [21], we define our cost as the L^1-norm of the ℓ_p-norm of the control. The best choice for $p \geq 1$ depends on the propulsion system geometry, and on the frame within which $\mathbf{u}(t) = \sum_{i=1}^{N} \mathbf{\Delta v}_i \delta(t - \tau_i)$ in J is resolved. Minimizing propellant directly requires resolving $\mathbf{\Delta v}_i$ into the spacecraft body-fixed frame; unfortunately, without relaxations, this requires spacecraft attitude $\mathbf{q}$ to be included in our state $\mathbf{x}$. To avoid this, a standard used throughout the literature and routinely in practical applications is to employ $p = 2$ so that each $\mathbf{\Delta v}_i$ is as short as possible, and optimally allocate the commanded $\mathbf{\Delta v}_i$ to thrusters later (online, once the attitude is known). Though this moves propellant minimization to a separate control allocation step (which we discuss in more detail in the Other Trajectory Constraints subsection), it greatly simplifies the problem in a practical way without neglecting attitude. Because the cost of Δv-allocation can only grow due to the need to satisfy torque constraints or impulse bounds (*e.g.*, necessitating counter-opposing thrusters to achieve the same net Δv-vector), we are in effect minimizing the best-case, unconstrained propellant use of the spacecraft. As we will show in our numerical experiments, this does not detract significantly from the technique; the coupling of J with $p = 2$ to the actual propellant use through the minimum control-effort thruster Δv allocation problem seems to promote low propellant-cost solutions. Hence (in practice) J serves as a good proxy to propellant use, with the added benefit of independence from propulsion system geometry.

Boundary Conditions

Sampling-based motion planners generally assume a known initial state $\mathbf{x}_{\text{init}}$ and time t_{init} from which planning begins (*e.g.*, the current state of the spacecraft), and define one or more goal regions $\mathcal{X}_{\text{goal}}$ to which guidance is sought. In this paper, we assume the chaser targets only one goal state $\mathbf{x}_{\text{goal}}^{\mathrm{T}} = \begin{bmatrix} \delta\mathbf{r}_{\text{goal}}^{\mathrm{T}} & \delta\mathbf{v}_{\text{goal}}^{\mathrm{T}} \end{bmatrix}$ at a time ("exact convergence," $\mathcal{X}_{\text{goal}} = \{\mathbf{x}_{\text{goal}}\}$), where $\delta\mathbf{r}_{\text{goal}}$ is the goal position and $\delta\mathbf{v}_{\text{goal}}$ is the goal velocity. During numerical experiments, however, we sometimes permit termination at any state whose position and velocity lie within Euclidean balls $\mathcal{B}(\delta\mathbf{r}_{\text{goal}}, \epsilon_r)$ and $\mathcal{B}(\delta\mathbf{v}_{\text{goal}}, \epsilon_v)$, respectively ("inexact convergence," $\mathcal{X}_{\text{goal}} = \mathcal{B}(\mathbf{r}_{\text{goal}}, \epsilon_r) \times \mathcal{B}(\mathbf{v}_{\text{goal}}, \epsilon_v)$), where the notation $\mathcal{B}(\mathbf{r}, \epsilon) = \{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{r} - \mathbf{x}\| \leq \epsilon\}$ denotes a ball with center $\mathbf{r}$ and radius ϵ.

System Dynamics

Because spacecraft proximity operations incorporate significant drift, spatially-dependent external forces, and changes on fast timescales, any realistic solution must obey dynamic constraints; we cannot assume straight-line trajectories. In this paper, we employ the classical Clohessy-Wiltshire-Hill (CWH) equations [22, 23] for impulsive linearized motion about a circular reference orbit at radius r_{ref} about an inverse-square-law gravitational attractor with parameter μ. This model provides a first-order approximation to a chaser spacecraft's motion relative to a rotating target-centered coordinate system (see Fig. 1). The linearized equations of motion for this scenario as resolved in the Local Vertical, Local Horizontal (LVLH) frame of the target are given by:

$$\delta\ddot{x} - 3n_{\text{ref}}^2 \delta x - 2n_{\text{ref}} \delta\dot{y} = \frac{F_x}{m} \tag{2a}$$

$$\delta\ddot{y} + 2n_{\text{ref}} \delta\dot{x} = \frac{F_y}{m} \tag{2b}$$

$$\delta\ddot{z} + n_{\text{ref}}^2 \delta z = \frac{F_z}{m} \tag{2c}$$

where $n_{\text{ref}} = \sqrt{\frac{\mu}{r_{\text{ref}}^3}}$ is the orbital frequency (mean motion) of the reference spacecraft orbit, m is the spacecraft mass, $\mathbf{F} = [F_x, F_y, F_z]$ is some applied force, and $(\delta x, \delta y, \delta z)$ and $(\delta\dot{x}, \delta\dot{y}, \delta\dot{z})$ represent the cross-track ("radial"), in-track, and out-of-plane relative position and relative velocity, respectively. The CWH model is quite common, and used often for rendezvous and proximity operations in Low Earth Orbit (LEO) and for leader-follower formation flight dynamics.

Defining the state $\mathbf{x}$ as $[\,\delta x, \delta y, \delta z, \delta\dot{x}, \delta\dot{y}, \delta\dot{z}\,]^{\mathrm{T}}$ and the

applied force-per-unit-mass $\mathbf{u}$ as $\frac{1}{m}\mathbf{F}^{\mathrm{T}}$, the CWH equations can be described by the *linear time-invariant* (LTI) system:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t) = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{3}$$

where the dynamics matrix $\mathbf{A}$ and input matrix $\mathbf{B}$ are given by:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3n_{\mathrm{ref}}^2 & 0 & 0 & 0 & 2n_{\mathrm{ref}} & 0 \\ 0 & 0 & 0 & -2n_{\mathrm{ref}} & 0 & 0 \\ 0 & 0 & -n_{\mathrm{ref}}^2 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

As for any LTI system, we can express the solution to Eq. (3) for any time $t \geq t_0$ using superposition and the convolution integral as $\mathbf{x}(t) = e^{\mathbf{A}(t-t_0)}\mathbf{x}(t_0) + \int_{t_0}^{t} e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)\,\mathrm{d}\tau$. The expression $\mathbf{\Phi}(t,\tau) \triangleq e^{\mathbf{A}(t-\tau)}$ is called the *state transition matrix*, which importantly provides an analytical mechanism for computing state trajectories that we rely heavily upon in simulations. Note, throughout this work, we shall sometimes represent $\mathbf{\Phi}(t,\tau)$ as $\mathbf{\Phi}$ for brevity when its arguments are understood.

We now specialize the above to the case of N impulsive velocity changes at times $t_0 \leq \tau_i \leq t_{\mathrm{f}}$, for $i \in [1, \ldots, N]$, in which case $\mathbf{u}(\tau) = \sum_{i=1}^{N} \mathbf{\Delta v}_i \delta(\tau - \tau_i)$, where $\delta(y) = \{1 \text{ where } y = 0, \text{ or } 0 \text{ otherwise}\}$ signifies the Dirac-delta distribution. Substituting for $\mathbf{\Phi}$ and $\mathbf{u}(\tau)$, this yields:

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{\Phi}(t,t_0)\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{\Phi}(t,\tau)\mathbf{B}\left(\sum_{i=1}^{N} \mathbf{\Delta v}_i \delta(\tau - \tau_i)\right) \mathrm{d}\tau \\ &= \mathbf{\Phi}(t,t_0)\mathbf{x}(t_0) + \sum_{i=1}^{N} \int_{t_0}^{t} \mathbf{\Phi}(t,\tau)\mathbf{B}\mathbf{\Delta v}_i \delta(\tau - \tau_i)\,\mathrm{d}\tau, \end{aligned}$$

where on the second line we used the linearity of the integral operator. By the sifting property of δ, denoting N_t as the number of burns applied from t_0 up to time t, we have for all times $t \geq t_0$ the following expression for the impulsive solution to Eq. (3):

$$\mathbf{x}(t) = \mathbf{\Phi}(t,t_0)\mathbf{x}(t_0) + \sum_{i=1}^{N_t} \mathbf{\Phi}(t,\tau_i)\mathbf{B}\mathbf{\Delta v}_i \tag{5a}$$

$$= \mathbf{\Phi}(t,t_0)\mathbf{x}(t_0) + \underbrace{[\mathbf{\Phi}(t,\tau_1)\mathbf{B} \quad \ldots \quad \mathbf{\Phi}(t,\tau_{N_t})\mathbf{B}]}_{\triangleq \mathbf{\Phi}_v(t,\{\tau_i\}_i)} \underbrace{\begin{bmatrix} \mathbf{\Delta v}_1 \\ \vdots \\ \mathbf{\Delta v}_{N_t} \end{bmatrix}}_{\triangleq \mathbf{\Delta V}} \tag{5b}$$

$$= \mathbf{\Phi}(t,t_0)\mathbf{x}(t_0) + \mathbf{\Phi}_v(t,\{\tau_i\}_i)\mathbf{\Delta V}. \tag{5c}$$

Throughout this paper, the notations $\mathbf{\Delta V}$ for the stacked Δv-vector and $\mathbf{\Phi}_v(t,\{\tau_i\}_i)$ for the aggregated impulse state transition matrix (or simply $\mathbf{\Phi}_v$ for short, when the parameters t and $\{\tau_i\}_i$ are clear) implicitly imply only those burns i occurring before time t.

Obstacle Avoidance

Obstacle avoidance is imposed by requiring the spacecraft trajectory $\mathbf{x}(t)$ to stay within $\mathcal{X}_{\mathrm{free}}$ (or, equivalently, outside of the obstacle region $\mathcal{X}_{\mathrm{obs}}$) – typically a difficult non-convex constraint. For CWH proximity operations, $\mathcal{X}_{\mathrm{obs}}$ might include those states that result in a collision with a neighboring object, all positions which lie outside of a given approach corridor, all velocities violating a given relative guidance law, *etc*. In our numerical experiments, we assume $\mathcal{X}_{\mathrm{obs}}$ includes an ellipsoidal Keep Out Zone (KOZ) enclosing the target spacecraft centered at the origin and a conical nadir-pointing region that approximates its antenna beam pattern – this both enforces collision-avoidance and prevents the chaser from interfering with the target's communications.

Note that according to the definition of $\mathcal{X}_{\mathrm{free}}$, this also requires the solution $\mathbf{x}(t)$ to stay within the confines of $\mathcal{X}$ (CWH system dynamics do not guarantee that state trajectories will lie in $\mathcal{X}$ despite the fact that their endpoints do). Though not strictly necessary in practice, if $\mathcal{X}_{\mathrm{free}}$ is defined to mark the extent of reliable sensor readings or the boundary inside which CWH equation linearity assumptions hold, then this can be a useful constraint to enforce.

Other Trajectory Constraints

Many other types of constraints may be included to encode additional restrictions on state and control trajectories, which we represent here by a set of inequality constraints $\mathbf{g}$ and equality constraints $\mathbf{h}$ (note that $\mathbf{g}$ and $\mathbf{h}$ denote vector functions). To illustrate the flexibility of the sampling-based planning approach, we encode the following into constraints $\mathbf{g}$ (for brevity, we omit their exact representation, which is straightforward based on vector geometry):

$$T_{\mathrm{plan,min}} \leq t_{\mathrm{final}} - t_{\mathrm{init}} \leq T_{\mathrm{plan,max}} \tag{6}$$

$$\mathbf{\Delta v}_i \in \mathcal{U}(\mathbf{x}(\tau_i)) \quad \text{for all } i = [1, \ldots, N] \tag{7}$$

$$\bigcup_{k \in [1,\ldots,K]} \mathcal{P}_{ik}(-\mathbf{\Delta}\hat{\mathbf{v}}_{ik}, \beta_{\mathrm{plume}}, H_{\mathrm{plume}}) \cap \mathbb{S}_{\mathrm{target}} = \varnothing \quad \text{for all } i = [1, \ldots, N] \tag{8}$$

Here $0 \leq T_{\mathrm{plan,min}} < T_{\mathrm{plan,max}}$ represent minimum and maximum motion plan durations, $\mathcal{U}(\mathbf{x}(\tau_i))$ is the admissible *control set* corresponding to state $\mathbf{x}(\tau_i)$, $\mathcal{P}_{ik}$ is the exhaust plume emanating from thruster k of the chaser spacecraft while executing burn $\mathbf{\Delta v}_i$ at time τ_i, and $\mathbb{S}_{\mathrm{target}}$ is the target spacecraft circumscribing sphere. We motivate each constraint in turn.

Plan Duration Bounds—Plan duration bounds facilitate the inclusion of rendezvous windows based on the epoch of the chaser at $\mathbf{x}_{\mathrm{init}}(t_{\mathrm{init}})$; such windows might be determined by illumination requirements, grounds communication opportunities, or mission timing restrictions, for example. $T_{\mathrm{plan,max}}$ may also be used to ensure the errors incurred by our linearized CWH model, which grow with time, do not exceed acceptable tolerances.

Control Feasibility— Control set constraints are intended to encapsulate limitations on control authority imposed by propulsive actuators and their geometric distribution about the spacecraft. For example, given the maximum burn magnitude $0 < \Delta v_{\max}$, the constraint:

$$\|\mathbf{\Delta v}_i\|_2 \leq \Delta v_{\max} \text{ for all } i = [1, \ldots, N] \tag{9}$$

might be used to represent an upper bound on the impulse range achievable by a gimbaled thruster system that is able to direct thrust freely in all directions. In our case, we use $\mathcal{U}(\mathbf{x}(\tau_i))$ to represent all commanded net Δv-vectors that (i) satisfy the constraint Eq. (9) above, and also (ii) can be successfully allocated to thrusters along trajectory $\mathbf{x}(t)$ at time τ_i according to a simple minimum-control effort thruster allocation problem (a straightforward linear program (LP) [24]). To keep the paper self-contained, we repeat the problem here and in our notation. Let $\boldsymbol{\Delta}\mathbf{v}_i|_{\text{bf}}$ and $\mathbf{M}_i|_{\text{bf}}$ be the desired net Δv and moment vectors at burn time τ_i, resolved in the body-fixed frame according to attitude $\mathbf{q}(\tau_i)$ (we henceforth drop the bar, for brevity). Note the attitude $\mathbf{q}(\tau_i)$ must either be included in the state $\mathbf{x}(\tau_i)$ or be derived from it, as is assumed in this paper by imposing (along nominal trajectories) a nadir-pointing attitude profile for the chaser spacecraft. Let $\Delta v_{ik} = \|\boldsymbol{\Delta}\mathbf{v}_{ik}\|_2$ be the Δv-magnitude allocated to thruster k, which generates an impulse in direction $\boldsymbol{\Delta}\hat{\mathbf{v}}_{ik}$ at position $\boldsymbol{\rho}_{ik}$ from the spacecraft center-of-mass (both are constant vectors if resolved in the body-fixed frame). Finally, to account for the possibility of on or off thrusters, let η_{ik} be equal to 1 if thruster k is available for burn i, or 0 otherwise. Then the minimum-effort control allocation problem can be represented as:

$$
\begin{aligned}
\text{Given:} \quad & \text{On-off flags } \eta_{ik}, \text{thruster positions } \boldsymbol{\rho}_{ik}, \\
& \text{thruster axes } \boldsymbol{\Delta}\hat{\mathbf{v}}_{ik}, \text{commanded } \Delta v\text{-vector } \boldsymbol{\Delta}\mathbf{v}_i, \\
& \text{and commanded moment vector } \mathbf{M}_i \\
\underset{\Delta v_{ik}}{\text{minimize}} \quad & \sum_{k=1}^{K} \Delta v_{ik} \\
\text{subject to} \quad & \sum_{k=1}^{K} \boldsymbol{\Delta}\hat{\mathbf{v}}_{ik}(\eta_{ik}\Delta v_{ik}) = \boldsymbol{\Delta}\mathbf{v}_i \\
& \sum_{k=1}^{K} (\boldsymbol{\rho}_{ik} \times \boldsymbol{\Delta}\hat{\mathbf{v}}_{ik})(\eta_{ik}\Delta v_{ik}) = \mathbf{M}_i \\
& \Delta v_{\min,k} \leq \Delta v_{ik} \leq \Delta v_{\max,k}
\end{aligned}
\tag{10}
$$

where $\Delta v_{\min,k}$ and $\Delta v_{\max,k}$ represent minimum and maximum impulse limits on thruster k (due to actuator limitations, minimum impulse bit, pulse-width constraints, or maximum on-time restrictions, for example). Because Δv is directly-proportional to thrust through the Tsiokolvsky rocket equation, the formulation above is directly analogous to minimum-propellant consumption; as discussed in the Section 2 subsection, by using control trajectories that minimize commanded Δv-vector lengths $\|\boldsymbol{\Delta}\mathbf{v}_i\|$, we can drive propellant use downwards as much as possible subject to our thrust bounds and net torque constraints. In this work, we set $\mathbf{M}_i = \mathbf{0}$ to enforce torque-free burns and minimize the disturbance to our assumed attitude trajectory $\mathbf{q}(t)$.

Note we do not consider a minimum norm constraint in Eq. (9) for $\boldsymbol{\Delta}\mathbf{v}_i$. As previously discussed, $\|\boldsymbol{\Delta}\mathbf{v}_i\|$ is only a proxy for the true propellant cost computed from the thrust allocation problem (Eq. (10)). The value of the norm bound $\Delta v_{\max}$ may be computed from the thruster limits $\Delta v_{\min,k}, \Delta v_{\max,k}$ and knowledge of the thruster configuration.

Plume Impingement—Impingement of thruster exhaust on neighboring spacecraft can lead to dire consequences, including destabilizing effects on attitude caused by exhaust gas pressure, degradation of sensitive optical equipment and solar arrays, and unexpected thermal loading [25]. To take this into account, we generate representative exhaust plumes at the locations of each thruster firing. For burn i occurring at time τ_i, a right circular cone is generated with axis $-\boldsymbol{\Delta}\hat{\mathbf{v}}_{ik}$, half-angle β_{plume}, and height H_{plume} at each *active* thruster k ($\eta_{ik} = 1$) for which its allocated thrust Δv_{ik}^* is non-zero, as determined by the solution to Eq. (10). Intersections are checked with the target spacecraft circumscribing sphere, $\mathbb{S}_{\text{target}}$, which is used as a more efficiently-verified, conservative approximation to the exact target geometry. For an illustration, see Fig. 2.

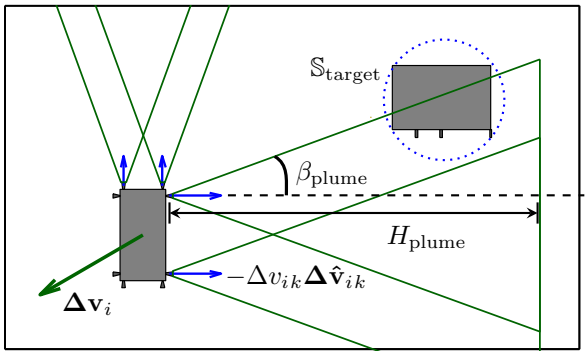


Figure 2. Illustration of exhaust plume impingement from thruster firings. Given a commanded $\boldsymbol{\Delta}\mathbf{v}_i$, the spacecraft must successfully allocate the impulse to thrusters while simultaneously avoiding impingement of neighboring object(s).

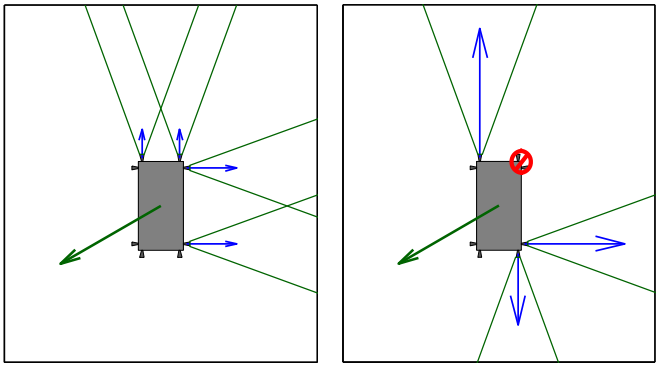
Other Constraints—Other constraints may easily be added. Solar array shadowing, pointing constraints, approach corridor constraints, *etc*., all fit within the framework, and may be represented as additional inequality or equality constraints. For additional examples, refer to [26].

Active Safety

An additional feature we include in our work is the concept of *active safety*, in which we require the target spacecraft to maintain a feasible Collision Avoidance Maneuver (CAM) to a safe higher or lower circular orbit from every point along its solution trajectory in the event that any mission-threatening control degradations such as stuck-off thrusters (as in Fig. 3) take place. We accomplish this by restricting our solution to pass through only those states that have a safe CAM available under all possible thruster failure configurations (up to a given thruster fault tolerance), with only coasting arcs (zero control effort trajectories) in between. This has been treated thoroughly in a previous work [27], from which the current paper derives. Because the design of our CAM policy is independent of the arrival time at a particular state and assuming our trajectory constraints are static (as can be shown for our numerical experiments), the safety of these states may be evaluated offline and cached — a fact we will make extensive use of in the design of our planning algorithm.

3. Planning Algorithm and Theoretical Characterization

With the proximity operations scenario established, we are now in position to describe our approach. As previously described, the constraints that must be satisfied in Eq. (1) are diverse, complex, and difficult to satisfy numerically. In this section, we propose a guidance algorithm to solve this problem, followed by a proof sketch of its optimality with regard to the 2-norm propellant-cost metric J under impulsive



(a) Thruster allocation without stuck-off failures.

(b) The same allocation problem, with both upper-right thrusters "stuck off."

Figure 3. Changes to torque-free control allocation in response to thruster failures. As can be seen, for the same net $\mathbf{\Delta v}$ vector (the large green arrow), thruster configuration changes can have a profound impact on thruster Δv locations and magnitudes (blue arrows), and hence on plume impingement satisfaction and thereby the safety of proposed $\mathbf{\Delta v}$ trajectories.

CWH dynamics. As will be seen, the algorithm relies on an understanding of: (i) the steering connections between sampled points assuming no obstacles or other trajectory constraints, and (ii) the nearest-neighbors or *reachable* states from a given state. We hence start by characterizing these two concepts, in the The Steering Problem and Reachability Sets subsections, respectively. We then proceed to the algorithm presentation in the Algorithm subsection, followed by its theoretical characterization in the Theoretical Characterization subsection.

The Steering Problem

In this section, we consider the *unconstrained* minimal propellant 2-point boundary value problem (2PBVP) or "steering problem" between an initial state $\mathbf{x}_0$ and a final state $\mathbf{x}_\mathrm{f}$ within the CWH dynamics model. Solutions to these steering problems provide the local building blocks from which we construct solutions to the more complicated problem formulation in Eq. (1). Steering solutions serve two main purposes: (i) they represent a class of short-horizon controlled trajectories that are filtered online for constraint satisfaction and efficiently strung together into a state space-spanning graph (*i.e.*, a tree or roadmap), and (ii) the costs of steering trajectories are used to inform the graph construction process by identifying the unconstrained "nearest neighbors" as edge candidates. Because these problems can be expressed independently of the arrival time t_0 (as will be shown), our solution algorithm does not need to solve these problems *online*; the solutions between every pair of samples can be precomputed and stored prior to receiving a motion query. Hence the 2PBVP presented here need not be solved quickly. However, we mention techniques here for speed-ups due to the reliance of our smoothing algorithm (Algorithm 2) on a fast solution method.

Substituting our boundary conditions into Eq. (5), evaluating at $t = t_\mathrm{f}$, and rearranging, we seek a stacked burn vector $\mathbf{\Delta V}$ such that:

$$\mathbf{\Phi}_v(t_\mathrm{f}, \{\tau_i\}_i)\mathbf{\Delta V} = \mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0, \tag{11}$$

for some number N of burn times $\tau_i \in [t_0,\, t_\mathrm{f}]$. Formulating this as an optimal control problem that minimizes our 2-norm cost functional (as a proxy for the actual propellant consumption, as described in the Cost Functional subsection of Section 2), we wish to solve:

$$\begin{aligned}
&\text{Given:} && \text{Initial state } \mathbf{x}_0\text{, final state } \mathbf{x}_\mathrm{f}\text{, burn norm}\\
&&& \text{bound } \Delta v_\mathrm{max}\text{, and duration bound } T_\mathrm{max}\\
&\underset{\mathbf{\Delta v}_i, \tau_i, t_\mathrm{f}, N}{\text{minimize}} && \sum_{i=1}^{N} \|\mathbf{\Delta v}_i\|_2\\
&\text{subject to} && \mathbf{\Phi}_v(t_\mathrm{f}, \{\tau_i\}_i)\mathbf{\Delta V} = \mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0.\\
&&& 0 \le t_\mathrm{f} - t_0 \le T_\mathrm{max}\\
&&& t_0 \le \tau_i \le t_\mathrm{f} \qquad \text{for burns } i\\
&&& \|\mathbf{\Delta v}_i\|_2 \le \Delta v_\mathrm{max} \qquad \text{for burns } i
\end{aligned} \tag{12}$$

Notice that this is a relaxed version of the original problem presented as Eq. (1), with only its boundary conditions, dynamic constraints, and control norm bound. As it stands, due to the nonlinearity of the dynamics with respect to τ_i, t_f and N, Eq. (12) is non-convex and inherently difficult to solve. However, we can make the problem tractable if we make a few assumptions. Given that we plan to string many steering trajectories together to form our overall solution, let us ensure they represent the most primitive building blocks possible such that their concatenation will adequately represent any arbitrary trajectory. Set $N = 2$ (the smallest number of burns required to transfer between any pair of arbitrary states, as it makes $\mathbf{\Phi}_v(t_\mathrm{f}, \{\tau_i\}_i)$ square) and select burn times $\tau_1 = t_0$ and $\tau_2 = t_\mathrm{f}$ (which automatically satisfy our burn time bounds). This leaves $\mathbf{\Delta v}_1 \in \mathbb{R}^{d/2}$ (an intercept burn applied just after $\mathbf{x}_0$ at time t_0), $\mathbf{\Delta v}_2 \in \mathbb{R}^{d/2}$ (a rendezvous burn applied just before $\mathbf{x}_\mathrm{f}$ at time t_f), and t_f as our only remaining decision variables. If we conduct a search for $t_\mathrm{f}^* \in [t_0,\, t_0 + T_\mathrm{max}]$, the relaxed-2PBVP can now be solved iteratively as a relatively simple bounded one-dimensional nonlinear minimization problem, where at each iteration one computes:

$$\mathbf{\Delta V}^*(t_\mathrm{f}) = \mathbf{\Phi}_v^{-1}(t_\mathrm{f}, \{t_0, t_\mathrm{f}\})(\mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0), \tag{13}$$

where the argument t_f is shown for $\mathbf{\Delta V}^*$ to highlight its dependence. By uniqueness of the matrix inverse (provided $\mathbf{\Phi}_v^{-1}$ is non-singular, discussed below), we need only check that the resulting impulses $\mathbf{\Delta v}_i^*(t_\mathrm{f})$ satisfy the magnitude bound to declare the solution to an iteration feasible. Notice that because $\mathbf{\Phi}$ and $\mathbf{\Phi}_v^{-1}$ depend only on the difference between t_f and t_0, we can equivalently search over various $t_\mathrm{f} - t_0 \in [0,\, T_\mathrm{max}]$ instead, using the expression:

$$\begin{aligned}
\mathbf{\Delta V}^*(t_\mathrm{f} - t_0) = \mathbf{\Phi}_v^{-1}(t_\mathrm{f} - t_0, \{0, t_\mathrm{f} - t_0\}) \cdot\\
(\mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f} - t_0, 0)\mathbf{x}_0),
\end{aligned} \tag{14}$$

which reveals that our impulsive steering problem depends only on the maneuver duration $T = t_\mathrm{f} - t_0$ (provided $\mathbf{x}_\mathrm{f}$ and $\mathbf{x}_0$ are given). This will be indispensable for precomputation, as it allows steering trajectories to be generated and stored *offline*. Regarding singularities, our steering solution $\mathbf{\Delta V}^* = \arg\min_{t_\mathrm{f}}(\mathbf{\Delta V}^*(t_\mathrm{f} - t_0))$ requires that $\mathbf{\Phi}_v$ be invertible, *i.e.*, that $(t_\mathrm{f} - \tau_1) - (t_\mathrm{f} - \tau_2) = t_\mathrm{f} - t_0$ avoids certain values (such as zero and certain values longer than one period [28], including orbital period multiples) which we achieve by restricting T_max to be shorter than one orbital period. To handle $t_\mathrm{f} - t_0 = 0$ exactly, note a solution to the 2PBVP exists if and only if $\mathbf{x}_0$ and $\mathbf{x}_\mathrm{f}$ differ in velocity only; in such cases, we take this velocity difference as $\mathbf{\Delta v}_2^*$ (with $\mathbf{\Delta v}_1^* = \mathbf{0}$) to be the solution.

Reachability Sets

In keeping with Eq. (14), since $\mathbf{\Delta V}^* = \arg\min_T(\mathbf{\Delta V}^*(T))$ only depends on $\mathbf{x}_\mathrm{f}$ and $\mathbf{x}_0$, we henceforth refer to the cost of a steering trajectory by the notation $J(\mathbf{x}_0, \mathbf{x}_\mathrm{f})$. We then define the forward *reachability set* from a given state $\mathbf{x}_0$ as follows:

Definition 3.1 (Forward Reachable Set) The forward reachable set $\mathcal{R}$ from state $\mathbf{x}_0$ is the set of all states $\mathbf{x}_\mathrm{f}$ that can be reached from $\mathbf{x}_0$ with a cost $J(\mathbf{x}_0, \mathbf{x}_\mathrm{f})$ at or below a given cost threshold $\bar{J}$, *i.e.*,

$$\mathcal{R}\big(\mathbf{x}_0, \bar{J}\big) \triangleq \big\{\mathbf{x}_\mathrm{f} \in \mathcal{X} \;\big|\; J(\mathbf{x}_0, \mathbf{x}_\mathrm{f}) \leq \bar{J}\big\}. \tag{15}$$

Recall from Eq. (14) in the The Steering Problem subsection that the steering cost may be written as:

$$\begin{aligned} J(\mathbf{x}_0, \mathbf{x}_\mathrm{f}) &= \|\mathbf{\Delta v}_1\| + \|\mathbf{\Delta v}_2\| \\ &= \|\mathbf{S}_1\mathbf{\Delta V}\| + \|\mathbf{S}_2\mathbf{\Delta V}\| \end{aligned} \tag{16}$$

where $\mathbf{S}_1 = [\,\mathbf{I}_{d/2\times d/2}\;\mathbf{0}_{d/2\times d/2}\,]$, $\mathbf{S}_2 = [\,\mathbf{0}_{d/2\times d/2}\;\mathbf{I}_{d/2\times d/2}\,]$, and $\mathbf{\Delta V}$ is given by:

$$\begin{aligned} \mathbf{\Delta V}(\mathbf{x}_0, \mathbf{x}_\mathrm{f}) &\triangleq \begin{bmatrix}\mathbf{\Delta v}_1 \\ \mathbf{\Delta v}_2\end{bmatrix} \\ &= \mathbf{\Phi}_v^{-1}(t_\mathrm{f}, \{t_0, t_\mathrm{f}\})(\mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0). \end{aligned} \tag{17}$$

The cost function $J(\mathbf{x}_0, \mathbf{x}_\mathrm{f})$ is difficult to gain insight on directly; however, as we shall see, we can work with its bounds much more easily.

Lemma 3.2 (Fuel Burn Cost Bounds) For the cost function in Eq. (16), we have the following upper and lower bounds:

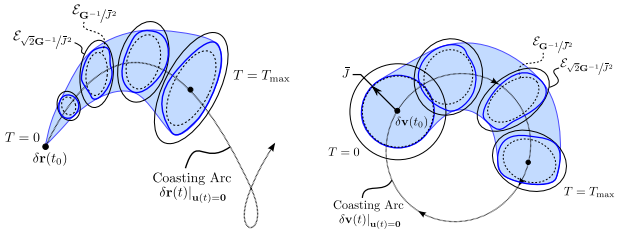
$$\|\mathbf{\Delta V}\| \leq J(\mathbf{x}_0, \mathbf{x}_\mathrm{f}) \leq \sqrt{2}\|\mathbf{\Delta V}\|. \tag{18}$$

Proof: For the upper bound, note that by the Cauchy-Schwarz inequality we have $J = \|\mathbf{\Delta v}_1\| \cdot 1 + \|\mathbf{\Delta v}_2\| \cdot 1 \leq \sqrt{\|\mathbf{\Delta v}_1\|^2 + \|\mathbf{\Delta v}_2\|^2} \cdot \sqrt{1^2 + 1^2}$. That is, $J \leq \sqrt{2}\|\mathbf{\Delta V}\|$. Similarly, for the lower bound, note that: $J = \sqrt{(\|\mathbf{\Delta v}_1\| + \|\mathbf{\Delta v}_2\|)^2} \geq \sqrt{\|\mathbf{\Delta v}_1\|^2 + \|\mathbf{\Delta v}_2\|^2} = \|\mathbf{\Delta V}\|$. ■

Now, observe that:

$$\|\mathbf{\Delta V}\| = \sqrt{(\mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0)^\mathrm{T}\mathbf{G}^{-1}(\mathbf{x}_\mathrm{f} - \mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0)},$$

where $\mathbf{G}^{-1} = \mathbf{\Phi}_v^{-\mathrm{T}}\mathbf{\Phi}_v^{-1}$, *i.e.*, the expression for an ellipsoid $\mathcal{E}(\mathbf{x}_\mathrm{f})$ resolved in the LVLH frame with matrix $\mathbf{G}^{-1}$ and center $\mathbf{\Phi}(t_\mathrm{f}, t_0)\mathbf{x}_0$ (the state $T = t_\mathrm{f} - t_0$ time units ahead of $\mathbf{x}_0$ along its coasting arc). Combined with Lemma 3.2, we see that for a fixed maneuver time T and propellant cost threshold $\bar{J}$, the spacecraft at $\mathbf{x}_0$ can reach all states inside an area under-approximated by an ellipsoid with matrix $\mathbf{G}^{-1}/\bar{J}^2$ and over-approximated by an ellipsoid of matrix $\sqrt{2}\mathbf{G}^{-1}/\bar{J}^2$. The forward reachable set for impulsive CWH dynamics under the 2-norm metric is therefore bounded by the union over all maneuver times of these under- and over-approximating ellipsoidal sets, respectively. See Fig. 4 for visualization.



(a) The set of reachable positions $\delta\mathbf{r}_\mathrm{f}$ within duration $T_{\max}$ and propellant cost $\Delta v_{\max}$.

(b) The set of reachable velocities $\delta\mathbf{v}_\mathrm{f}$ within duration $T_{\max}$ and propellant cost $\Delta v_{\max}$.

Figure 4. Visualizing reachability sets (shaded blue) from initial state $\mathbf{x}(t_0) = [\,\delta\mathbf{r}(t_0), \delta\mathbf{v}(t_0)\,]$ given propellant cost threshold $\bar{J}$. For cost measured as a sum of $\mathbf{\Delta v}$ 2-norms, these are bounded by unions over maneuver duration $T \in [0,\, T_{\max}]$ of ellipsoidal balls in position-velocity (phase) space centered about the spacecraft's coasting arc. For the special case of $T = 0$, the reachable set of states is a 3-dimensional velocity ball embedded in $\mathbb{R}^6$ (with volume 0) corresponding to states with position $\delta\mathbf{r}_\mathrm{f} = \delta\mathbf{r}(t_0)$ and velocity $\delta\mathbf{v}_\mathrm{f} \in \mathcal{B}\big(\delta\mathbf{v}(t_0), \bar{J}\big)$.

Algorithm

As mentioned in Section 1, we apply a modified version of the Fast Marching Tree (FMT*) sampling-based planning algorithm to solve the problem in Eq. (1). Sampling-based planning [14, 29] essentially breaks down a continuous trajectory optimization problem into a series of relaxed, local steering problems (as in The Steering Problem subsection) between intermediate waypoints (called *samples*) before piecing them together to form a global solution to the original problem. This framework can yield significant computational benefits if: (i) the relaxed subproblems are simple enough, and (ii) the *a posteriori* evaluation of trajectory constraints is fast compared to a single solution of the full-scale problem. Furthermore, provided samples are sufficiently dense in the free state-space $\mathcal{X}_\mathrm{free}$ and graph exploration is spatially symmetric, sampling-based planners can closely approximate global optima without fear of convergence to local minima. Though many candidate planners could be used here, we rely on the asymptotically-optimal (AO) FMT* algorithm for its efficiency (see [20] for details on the advantages of FMT* over its state-of-the-art counterparts) and its compatibility with *deterministic* (as opposed to random) sampling sequences [30], which leads to a number of algorithmic simplifications (including use of offline knowledge).

The FMT* algorithm, tailored to our application, is presented as Algorithm 1 (we shall henceforth refer to our modified version of FMT* as simply FMT*, for brevity). FMT* efficiently expands a tree of feasible trajectories from an initial state $\mathbf{x}_\mathrm{init}$ to a goal state $\mathbf{x}_\mathrm{goal}$ around nearby obstacles. It begins by taking a set of samples distributed in the free state space $\mathcal{X}_\mathrm{free}$ using the SAMPLEFREE routine, which restricts state sampling to actively-safe, collision-free samples (which lie outside of $\mathcal{X}_\mathrm{obs}$ and have access to a safe Collision-Avoidance Maneuver (CAM) as described in the Active Safety subsection of Section 2). In our implementation, we assume samples are taken using a particular deterministic, low-dispersion sequence called the Halton sequence [31], though any deterministic, *low-dispersion* sampling sequence may be used [30]. Selecting $\mathbf{x}_\mathrm{init}$ first for further expansion as the minimum cost-to-come node $\mathbf{z}$, the algorithm then proceeds to look at reachable samples or "neighbors" (samples that can be reached with less than a given propellant cost threshold $\bar{J}$, as described in the previous subsection) and attempts connections (using STEER, as described in The Steering Problem subsection) to those with cheapest cost-to-come back to the tree. The cost threshold

$\bar{J}$ is a free parameter whose value can have a significant effect on performance; see Theorem 3.5 for a theoretical characterization and Section 5 for a representative numerical trade study. Those trajectories satisfying the constraints of Eq. (1), as determined by COLLISIONFREE, are saved. As feasible connections are made, the algorithm relies on adding and removing nodes (saved waypoint states) from three sets: a set of unexplored samples $\mathcal{V}_{\text{unvisited}}$ not yet connected to the tree, a frontier $\mathcal{V}_{\text{open}}$ of nodes likely to make efficient connections to unexplored neighbors, and an interior $\mathcal{V}_{\text{closed}}$ of nodes that are no longer useful for exploring the state space $\mathcal{X}$. Details on FMT* can be found in its original work [20].

To make FMT* amenable to a real-time implementation, we consider an online-offline approach that relegates as much computation as possible to a pre-processing phase. To be specific, the sample set $\mathcal{S}$ (line 2), nearest-neighbor sets (used in lines 5 and 6), and steering trajectory solutions (line 7) may be entirely pre-processed, assuming the planning problem satisfies the following conditions:

1. The state space $\mathcal{X}$ is known *a priori*, as is typical for most LEO missions (a luxury we do not generally have for the obstacle space $\mathcal{X}_{\text{obs}}$, which must be identified online using onboard sensors once the spacecraft arrives at $\mathcal{X}$),
2. Steering solutions are independent of sample arrival times t_0, as we showed in The Steering Problem subsection.

Here Item 1 allows samples to be precomputed, while Item 2 enables steering trajectories to be stored onboard or uplinked from the ground up to the spacecraft, since their values remain relevant regardless of the times at which the spacecraft actually follows them during the mission. This leaves only collision-checking, graph construction, and termination checks as parts of the online phase, greatly improving the online run time and leaving the more intensive work to offline resources where running time is less important. This breakdown into online and offline components (inspired by [32]) is a valuable technique for imbuing kinodynamic motion planning problems with real-time online solvability using fast batch-planners like FMT*.

Theoretical Characterization

It remains to show that FMT* provides similar asymptotic optimality and convergence rate guarantees under the 2-norm propellant-cost metric and impulsive CWH dynamics (which enter into Algorithm 1 under lines 6–7), as it does for kinematic (straight-line path planning) problems [20]. For sampling-based algorithms, *asymptotic optimality* refers to the property that as the number of samples $n \to \infty$, the cost of the trajectory (*a.k.a.* "path") returned by the planner approaches that of the optimal cost. Here a proof sketch is presented showing asymptotic optimality for the planning algorithm and problem setup used in this paper. Full details may be found in [33]. We note that while CWH dynamics are the primary focus of this work, the following proof methodology extends to any general linear system controlled by a finite sequence of impulsive actuations, whose fixed-duration 2-impulse steering problem is uniquely determined (*e.g.*, a wide array of second-order control systems).

The proof proceeds analogously to [20] by showing that it is always possible to construct an approximate path from points in $\mathcal{S}$ that closely follows the optimal path. Similarly to [20], we will make use here of a concept called the ℓ_2-*dispersion* of a set of points, which upper bounds how far away a point in $\mathcal{X}$ can be from its nearest point in $\mathcal{S}$ as measured by the ℓ_2-norm.

Algorithm 1 The Fast Marching Tree Algorithm (FMT*). Computes a minimal-cost trajectory from an initial state $\mathbf{x}(t_0) = \mathbf{x}_{\text{init}}$ to a target state $\mathbf{x}_{\text{goal}}$ through a fixed number n of samples $\mathcal{S}$.

1: Add $\mathbf{x}_{\text{init}}$ to the root of the tree $\mathcal{T}$, as a member of the frontier set $\mathcal{V}_{\text{open}}$
2: Generate samples $\mathcal{S} \leftarrow \text{SAMPLEFREE}(\mathcal{X}, n, t_0)$ and add them to the unexplored set $\mathcal{V}_{\text{unvisited}}$
3: Set the minimum cost-to-come node in the frontier set as $\mathbf{z} \leftarrow \mathbf{x}_{\text{init}}$
4: **while true**
5: **for** each neighbor $\mathbf{x}$ of $\mathbf{z}$ in $\mathcal{V}_{\text{unvisited}}$
6: Find the neighbor $\mathbf{x}_{\min}$ in $\mathcal{V}_{\text{open}}$ of cheapest cost-to-go to $\mathbf{x}$
7: Compute the trajectory between them as $[\mathbf{x}(t), \mathbf{u}(t), t] \leftarrow \text{STEER}(\mathbf{x}_{\min}, \mathbf{x})$
8: **if** COLLISIONFREE$(\mathbf{x}(t), \mathbf{u}(t), t)$
9: Add the trajectory from $\mathbf{x}_{\min}$ to $\mathbf{x}$ to tree $\mathcal{T}$
10: Remove all $\mathbf{x}$ from the unexplored set $\mathcal{V}_{\text{unvisited}}$
11: Add any new connections $\mathbf{x}$ to the frontier $\mathcal{V}_{\text{open}}$
12: Remove $\mathbf{z}$ from the frontier $\mathcal{V}_{\text{open}}$ and add it to $\mathcal{V}_{\text{closed}}$
13: **if** $\mathcal{V}_{\text{open}}$ is empty
14: **return** Failure
15: Reassign $\mathbf{z}$ as the node in $\mathcal{V}_{\text{open}}$ with smallest cost-to-come from the root ($\mathbf{x}_{\text{init}}$)
16: **if** $\mathbf{z}$ is in the goal region $\mathcal{X}_{\text{goal}}$
17: **return** Success, and the unique trajectory from the root ($\mathbf{x}_{\text{init}}$) to $\mathbf{z}$

Definition 3.3 (ℓ_2-dispersion) For a finite, non-empty set $\mathcal{S}$ of points in a d-dimensional compact Euclidean subspace $\mathcal{X}$ with positive Lebesgue measure, its ℓ_2-dispersion $D(\mathcal{S})$ is defined as:

$$D(\mathcal{S}) \triangleq \sup_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{s}\in\mathcal{S}} \|\mathbf{s}-\mathbf{x}\| = \sup\{R > 0 \mid \exists \mathbf{x} \in \mathcal{X} \text{ with } \mathcal{B}(\mathbf{x}, R) \cap \mathcal{S} = \varnothing\}, \tag{19}$$

where $\mathcal{B}(\mathbf{x}, R)$ is a Euclidean ball with radius R centered at state $\mathbf{x}$.

In order to approximate a trajectory by points in $\mathcal{S}$, we also need a means for quantifying the obstacle-free space around the trajectory.

Definition 3.4 (Strong δ-Clearance) A trajectory $\mathbf{x}(t)$ is said to have *strong* δ-clearance if, for some $\delta > 0$ and all t, the Euclidean distance between $\mathbf{x}(t)$ and any point in $\mathcal{X}_{\text{obs}}$ is greater than δ.

We are now in a position to show that the cost of the trajectory returned by FMT* approaches that of an optimal trajectory as the number of samples $n \to \infty$. The proof proceeds in two steps. First, we establish that there is a sequence of waypoints in $\mathcal{S}$ that are placed closely along the optimal path and approximately evenly-spaced in cost. Then we note that the existence of these waypoints guarantees that FMT* finds a path with a cost close to that of the optimal cost.

Theorem 3.5 (Asymptotic Performance of FMT*) Let $\mathbf{x}^*(t)$ be a feasible trajectory satisfying Eq. (1) with strong δ-clearance and cost J^*. Let $\mathcal{S} \cup \{\mathbf{x}_0\}$ be a set of $n \in \mathbb{N}$ samples

from $\mathcal{X}_{\text{free}}$ with dispersion $D(\mathcal{S}) \leq \gamma n^{-1/d}$. Finally, let J_n denote the cost of the path returned by FMT* with n points in $\mathcal{S}$ while using a cost threshold $\bar{J}(n) = \omega\left(n^{-1/3d}\right)$ and $\bar{J} = o(1)$. (That is, $\bar{J}(n)$ asymptotically dominates $n^{-1/3d}$ and is asymptotically dominated by 1.) Then $\lim_{n\to\infty} J_n \leq J^*$.

Proof Sketch: The basis of the argument is that small endpoint pertubations in the 2-impulse 2PBVP bring about correspondingly small deviations in the steering control. This fact ensures that the particular placement of the points of $\mathcal{S}$ is immaterial; only its low-dispersion property matters. We consider states $\{\mathbf{x}_k\}$ equally spaced along $\mathbf{x}^*(t)$ in cost intervals of $\bar{J}(n)/2$. We claim that we can find a sequence of points $\{\mathbf{y}_k\} \subset \mathcal{S} \cup \{\mathbf{x}_0\}$, which may be thought of as small perturbations on the $\{\mathbf{x}_k\}$, such that by concatenating steering connections between the $\{\mathbf{y}_k\}$ we construct a trajectory with cost approaching J^* and which stays within the obstacle-free envelope afforded by the δ-clear property of $\mathbf{x}^*(t)$.

In contrast with previous perturbation results for linear systems [34], the existence of zero-time trajectory segments brought about by $\mathbf{x}^*(t)$ impulse controls prevents us from selecting the $\{\mathbf{y}_k\}$ as symmetric perturbations on the $\{\mathbf{x}_k\}$. Instead of selecting each $\mathbf{y}_k$ as a point within a ball centered at $\mathbf{x}_k$, we consider a shifted center $\tilde{\mathbf{x}}_k$ for each ball. In the case that the trajectory segment from $\mathbf{x}_{k-1}$ to $\mathbf{x}_k$ in $\mathbf{x}^*(t)$ is contained within an instantaneous impulse burn, the position of the shifted center $\tilde{\mathbf{x}}_k$ incorporates the drift incurred over the short (but nonzero) time required to correct for the arbitary placement of $\mathbf{y}_k$ within a perturbation ball of radius proportional to $\bar{J}(n)^3$ (which exceeds the dispersion $D(\mathcal{S})$ for sufficiently large n). Bounds on the position shifts of the $\{\tilde{\mathbf{x}}_k\}$, as well as the resulting cost of steering from each $\mathbf{y}_{k-1}$ to $\mathbf{y}_k$, may be established which ensure that the $\{\mathbf{y}_k\}$ are clear of obstacles, successively separated in cost by at most $\bar{J}(n)$, and delineate a trajectory with cost approaching J^* as $n \to \infty$. The existence of this sequence of waypoints $\{\mathbf{y}_k\}$ for each n as a candidate path for recovery by FMT* is sufficient to establish that $\lim_{n\to\infty} J_n \leq J^*$. ■

Remark 3.6 (Asymptotic Optimality of FMT*) If the planning problem at hand admits an optimal solution that does not itself have strong δ-clearance, but is arbitrarily approximable both pointwise and in cost by trajectories with strong clearance (see [35] for additional discussion on why such an assumption is reasonable), then Theorem 3.5 implies the asymptotic optimality of FMT*.

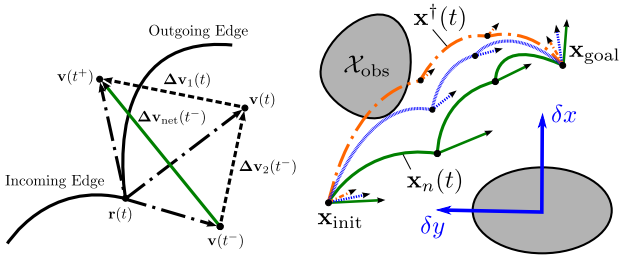
4. Trajectory Smoothing

Due to the discreteness caused by using a *finite* number of samples, sampling-based solutions will necessarily be approximations to true optima. In an effort to compensate for this limitation, we offer in this section two techniques to improve the quality of solutions returned by our planner from the Algorithm subsection of Section 3. We first describe a straightforward method for reducing the sum of Δv-vector magnitudes along concatenated sequences of edge trajectories that can also be used to improve the search for propellant-efficient trajectories in the feasible state space $\mathcal{X}_{\text{free}}$. We then follow with a fast post-processing algorithm for further reducing propellant cost after a solution has been reported.

The first technique removes unnecessary Δv-vectors that occur when joining sub-trajectories (edges) in the planning graph. Consider merging two edges at a node with position $\delta\mathbf{r}(t)$ and velocity $\delta\mathbf{v}(t)$ as in Fig. 5(a). A naive concatenation would retain both $\mathbf{\Delta v}_2(t^-)$ (the rendezvous burn added to the incoming velocity $\mathbf{v}(t^-)$) and $\mathbf{\Delta v}_1(t)$ (the intercept burn used to achieve the outgoing velocity $\mathbf{v}(t^+)$) individually within the combined control trajectory. Yet, because these impulses occur at the same time, a more realistic approach should merge them into a single net Δv-vector $\mathbf{\Delta v}_{\text{net}}(t^-)$. By the triangle inequality, we have that:

$$\begin{aligned} \left\|\mathbf{\Delta v}_{\text{net}}(t^-)\right\| &= \left\|\mathbf{\Delta v}_2(t^-) + \mathbf{\Delta v}_1(t)\right\| \\ &\leq \left\|\mathbf{\Delta v}_2(t^-)\right\| + \left\|\mathbf{\Delta v}_1(t)\right\|. \end{aligned} \tag{20}$$

Hence, merging edges in this way guarantees Δv savings for solution trajectories under the 2-norm propellant metric. Furthermore, incorporating net Δv's into the cost-to-come during graph construction can make exploration of the search space more efficient; the cost-to-come $c(\mathbf{z})$ for a given node $\mathbf{z}$ would then reflect the cost to rendezvous with $\mathbf{z}$ from $\mathbf{x}_{\text{init}}$ through a series of intermediate intercepts rather than a series of rendezvous maneuvers (as a trajectory designer might normally expect). Note, on the other hand, that two edges as in Fig. 5(a) that are merged in this fashion no longer achieve velocity $\mathbf{v}(t)$; state $\mathbf{x}(t)$ is skipped altogether. This may not be desirable in certain applications (*e.g.*, for some active safety policies) which rely on rendezvousing with intermediate waypoints like $\mathbf{x} = [\,\mathbf{r}(t)\ \mathbf{v}(t)\,]$ *exactly*.



(a) Smoothing during graph construction (merges Δv-vectors at edge endpoints).

(b) Smoothing during post-processing (see Algorithm 2).

Figure 5. Improving sampling-based solutions under minimal-propellant impulsive dynamics. Figure 5(a) can be used to merge Δv-vectors between edge endpoints during and after graph construction, while Fig. 5(b) illustrates the post-processing smoothing algorithm given in Algorithm 2 (the original trajectory $\mathbf{x}_n(t)$ is solid, the approximate unconstrained optimum $\mathbf{x}^\dagger(t)$ is dash-dotted, and the resulting "smoothed" trajectory derived from their combination is shown dashed).

The second technique attempts to reduce solution cost by adjusting the magnitudes of Δv-vectors in the trajectory returned by FMT*, denoted by $\mathbf{x}_n(t)$ with associated stacked impulse vector $\mathbf{\Delta V}_n$. By relaxing FMT*'s constraint to pass through state samples, strong cost improvements may be gained. The main idea is to deform our low-cost, feasible solution $\mathbf{x}_n(t)$ as much as possible towards the unconstrained minimum-propellant solution $\mathbf{x}^*(t)$ between $\mathbf{x}_{\text{init}}$ and $\mathbf{x}_{\text{goal}}$, as determined by the 2-point Boundary Value Problem (Eq. (12)) solution from Section 3 (in other words, use a homotopic transformation from $\mathbf{x}_n(t)$ to $\mathbf{x}^*(t)$). However, a naive attempt to solve Eq. (12) in its full generality would be too time-consuming to be useful, and would threaten the real-time capability of our approach. Assuming our sampling-based trajectory is near-optimal (or at least, in a low-cost solution homotopy), we can relax Eq. (12) by keeping the number of burns N, end time $t_{\text{f}} := t_{\text{final}}$, and burn times τ_i fixed from our planning solution, and solve for

an approximate unconstrained minimum-propellant solution $\mathbf{\Delta V}^\dagger$ with associated state trajectory $\mathbf{x}^\dagger(t)$ via:

$$\begin{aligned} \underset{\mathbf{\Delta v}_i}{\text{minimize}} \quad & \sum_{i=1}^{N} \|\mathbf{\Delta v}_i\|_2 \\ \text{subject to} \quad & \mathbf{\Phi}_v(t_{\text{final}}, \{\tau_i\}_i)\mathbf{\Delta V} = \mathbf{x}_{\text{goal}} - \mathbf{\Phi}(t_{\text{final}}, t_{\text{init}})\mathbf{x}_{\text{init}} \\ & \|\mathbf{\Delta v}_i\|_2 \leq \Delta v_{\max} \quad \text{for all burns } i \end{aligned} \tag{21}$$

(see Section 2 for definitions). It can be shown that Eq. (21) is a second-order cone program (SOCP), and hence quickly solved using standard convex solvers. As the following proof shows explicitly, we can safely deform the trajectory $\mathbf{x}_n(t)$ towards $\mathbf{x}^\dagger(t)$ without violating our dynamics and boundary conditions if we use a convex combination of our two control trajectories $\mathbf{\Delta V}_n$ and $\mathbf{\Delta V}^\dagger$. This follows from the principle of superposition, given that the CWH equations are Linear, Time-Invariant (LTI), and the fact that both solutions already satisfy the boundary conditions.

Theorem 4.1 (Dynamic Feasibility of CWH Smoothing) Suppose $\mathbf{x}_n(t)$ and $\mathbf{x}^\dagger(t)$ with respective control vectors $\mathbf{\Delta V}_n$ and $\mathbf{\Delta V}^\dagger$ are two state trajectories which satisfy the impulsive CWH steering problem Eq. (11) between states $\mathbf{x}_{\text{init}}$ and $\mathbf{x}_{\text{goal}}$. Then the trajectory $\mathbf{x}(t)$ generated by the convex combination of $\mathbf{\Delta V}_n$ and $\mathbf{\Delta V}^\dagger$ is itself a convex combination of $\mathbf{x}_n(t)$ and $\mathbf{x}^\dagger(t)$, and hence also satisfies Eq. (11).

Proof: Let $\mathbf{\Delta V} = \alpha\mathbf{\Delta V}_n + (1-\alpha)\mathbf{\Delta V}^\dagger$ for some value $\alpha \in [0,\, 1]$. From our dynamics equation,

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{\Phi}(t, t_{\text{init}})\mathbf{x}_{\text{init}} + \mathbf{\Phi}_v(t, \{\tau_i\}_i)\mathbf{\Delta V} \\ &= [\alpha + (1-\alpha)]\mathbf{\Phi}(t, t_{\text{init}})\mathbf{x}_{\text{init}} \\ &\quad + \mathbf{\Phi}_v(t, \{\tau_i\}_i)\Big[\alpha\mathbf{\Delta V}_n + (1-\alpha)\mathbf{\Delta V}^\dagger\Big] \\ &= \alpha[\mathbf{\Phi}(t, t_{\text{init}})\mathbf{x}_{\text{init}} + \mathbf{\Phi}_v(t, \{\tau_i\}_i)\mathbf{\Delta V}_n] \\ &\quad + (1-\alpha)\Big[\mathbf{\Phi}(t, t_{\text{init}})\mathbf{x}_0 + \mathbf{\Phi}_v(t, \{\tau_i\}_i)\mathbf{\Delta V}^\dagger\Big] \\ &= \alpha\mathbf{x}_n(t) + (1-\alpha)\mathbf{x}^\dagger(t) \end{aligned}$$

which is a convex combination, as required. Substituting $t = t_{\text{init}}$ or $t = t_{\text{goal}}$, we see that $\mathbf{x}(t)$ satisfies the boundary conditions given that $\mathbf{x}_n(t)$ and $\mathbf{x}^\dagger(t)$ do. This completes the proof. ■

We take advantage of this fact for trajectory-smoothing. Our algorithm, reported as Algorithm 2 and illustrated in Fig. 5(b), computes the approximate unconstrained minimum-propellant solution $\mathbf{x}^\dagger(t)$ and returns it (if feasible) or otherwise conducts a bisection line search on α, returning a convex combination of our original planning solution $\mathbf{x}_n(t)$ and $\mathbf{x}^\dagger(t)$ that comes as close to $\mathbf{x}^\dagger(t)$ as possible without violating trajectory constraints. Note because $\mathbf{\Delta V}_n$ lies in the feasible set of Eq. (21), the algorithm can only improve the final propellant cost. By design, Algorithm 2 is geared towards reducing our original solution propellant-cost as quickly as possible while maintaining feasibility; the most expensive computational components are the calculation of $\mathbf{\Delta V}^\dagger$ and collision-checking (consistent with our sampling-based algorithm). Fortunately, the number of collision-checks is limited by the maximum number of iterations $\left\lceil \log_2\left(\frac{1}{\delta\alpha_{\min}}\right)\right\rceil + 1$, given tolerance $\delta\alpha_{\min} \in (0,\, 1)$. As an added bonus, for strictly time-constrained applications that require a solution in a fixed amount of time, the algorithm can be easily modified to return the α_ℓ-weighted trajectory $\mathbf{x}_{\text{smooth}}(t)$ when time runs out, as the feasibility of this trajectory is maintained as an algorithm invariant.

Algorithm 2 "Trajectory smoothing" algorithm for impulsive CWH dynamics. Given a trajectory $\mathbf{x}_n(t), t \in [t_{\text{init}},\, t_{\text{goal}}]$ between initial and goal states $\mathbf{x}_{\text{init}}$ and $\mathbf{x}_{\text{goal}}$ satisfying Eq. (1) with impulses $\mathbf{\Delta V}_n$ applied at times $\{\tau_i\}_i$, returns another feasible trajectory with reduced 2-norm propellant-cost.

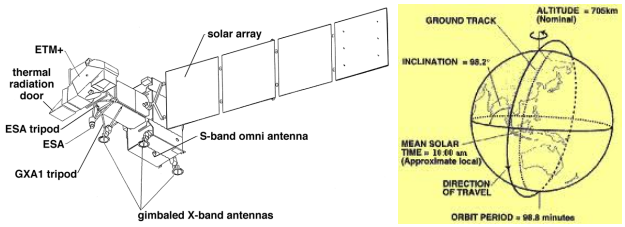
```
Initialize the smoothed trajectory x_smooth(t) as x_n(t), with ΔV_smooth = ΔV_n
Compute the unconstrained optimal control vector ΔV† by solving Eq. (21)
Compute the unconstrained optimal state trajectory x†(t) using Eq. (5) (See Section 2)
Initialize weight α and its lower and upper bounds as α ← 1, α_ℓ ← 0, α_u ← 1
while true
    x(t) ← (1 − α)x_n(t) + αx†(t)
    ΔV ← (1 − α)ΔV_n + αΔV†
    if CollisionFree(x(t), ΔV, t)
        α_ℓ ← α
        Save the smoothed trajectory x_smooth(t) as x(t)
        and control ΔV_smooth as ΔV
    else
        α_u ← α
    if α_u − α_ℓ is less than tolerance δα_min ∈ (0, 1)
        break
    α ← (α_ℓ + α_u)/2
return smoothed trajectory x_smooth(t), with ΔV_smooth
```

5. Numerical Experiments

Consider the scenario shown in Fig. 7, here modeling the near-field approach of a chaser spacecraft in close proximity[2] to a target moving in a circular LEO trajectory (as in Fig. 1). We imagine the chaser, which starts in a circular orbit of lower radius, must be repositioned through a sequence of pre-specified CWH waypoints (*e.g.*, required for equipment checks, surveying, *etc.*) to a coplanar position located radially above the target, arriving with zero relative velocity in preparation for a final radial ("R-bar") approach. Throughout the maneuver, as described in detail in Section 2, the chaser must avoid entering the elliptic target KOZ, enforce hard safety constraints with regard to a two-fault tolerance to stuck-off thruster failures, and otherwise avoid interfering with the target. This includes avoiding the target's nadir-pointing communication lobes (represented by truncated half-cones), and preventing exhaust plume impingement on its surfaces. For context, we use the Landsat-7 spacecraft and orbit as a reference [36, 3.2] (see Fig. 6).

If we take the waypoints in the guidance sequence one at a time as individual goal points $\mathbf{x}_{\text{goal}}$, we can solve the given scenario as a series of motion planning problems (or "subplans"),

[2] *Close proximity* in this context implies that any higher-order terms of the linearized relative dynamics are negligible, *e.g.*, within a few percent of the target orbit mean radius.



(a) Landsat-7 schematic (Nadir (-δx direction) points down, while the in-track ($+\delta y$) direction points left).

(b) Landsat-7 orbit (Courtesy of the Landsat-7 Handbook).

Figure 6. Target spacecraft geometry and orbital scenario used in numerical experiments.

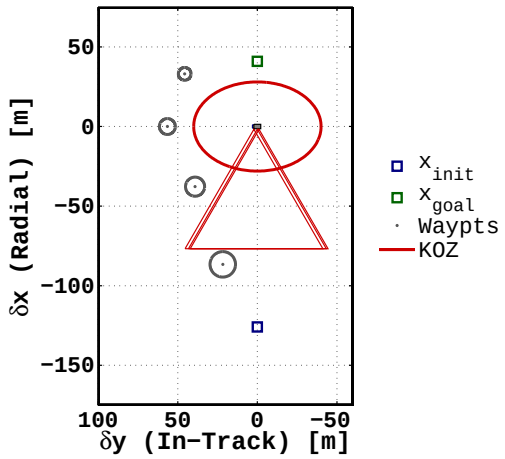
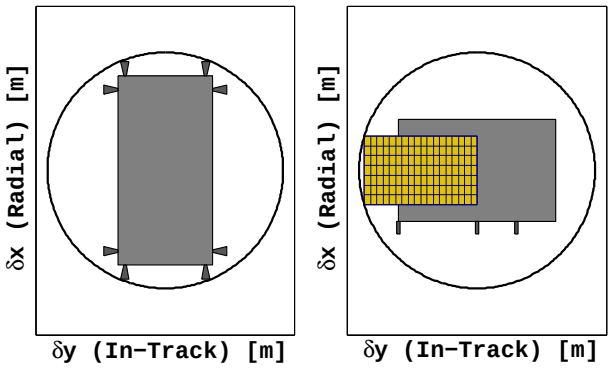


Figure 7. Illustration of the motion planning query in the LVLH frame. The spacecraft must track a series of guidance waypoints to the final goal state, which is located radially above the client. Positional goal tolerances are visualized as circles around each waypoint, which successively shrink in size.

calling FMT* from Section 3 once for each instance, linking them together to form an overall solution to the problem. As our steering controller from Section 3 is attitude-independent, we use position-velocity states $\mathbf{x} \in \mathbb{R}^d$ with $d = 4$, where $\mathbf{x} = [\,\delta x, \delta y, \delta \dot{x}, \delta \dot{y}\,]^{\mathrm{T}}$. We omit the attitude $\mathbf{q}$ from the state during planning by assuming the existence of an attitude policy (as well as a stable attitude-trajectory-following controller) that produces $\mathbf{q}(t)$ from the state trajectory $\mathbf{x}(t)$; for illustration purposes, a simple nadir-pointing attitude profile is chosen to represent a mission that requires constant communication with the ground throughout the maneuver (note this is not enforced along actively-safe escape trajectories, which for each failure mode execute a simple "turn-burn-turn" policy that orients the closest available thruster as quickly as possible in the direction required to implement the necessary circularization burn– see [27] for full details). Given the hyper-rectangular shape of the state-space, we call upon the deterministic, low-dispersion d-dimensional Halton sequence [31] to sample positions and velocities. To improve sample densities, each subplan uses its own sample space defined around only its respective initial and goal waypoints, with some arbitrary threshold space added around them. Additionally, extra samples n_{goal} are taken inside each waypoint ball to facilitate convergence. For this multi-plan problem, we define the solution cost as the sum of individual subplan costs (if using trajectory smoothing, the endpoints between two plans will be merged identically to two edges within a plan, as described in Section 4).

Before we proceed to the results, we make note of a few



(a) Chaser spacecraft

(b) Target spacecraft

Figure 8. Schematics of the chaser and target, together with their circumscribing spheres.

Table 1. List of parameters used during numerical experiments.

Chaser plume half-angle, β_{plume}	10°
Chaser plume height, H_{plume}	16 m
Chaser thruster fault tolerance, F	2
Cost threshold, $\bar{J}$	0.1–0.4 m/s
Dimension, d	4
Goal sample count, n_{goal}	$0.04n$
Goal position tolerance, ϵ_r	3–8 m
Goal velocity tolerance, ϵ_v	0.1–0.5 m/s
Max. thruster Δv norm, $\Delta v_{\mathrm{max},k}$	∞ m/s
Max. net Δv norm, Δv_{max}	∞ m/s
Max. plan duration, $T_{\mathrm{plan,max}}$	∞ s
Min. plan duration, $T_{\mathrm{plan,min}}$	0 s
Max. steering duration, T_{max}	$0.1 \cdot (\,2\pi/n_{\mathrm{ref}}\,)$
Min. steering duration, T_{min}	0 s
Sample count, n	50–400 per plan
Simulation timestep, Δt	$0.0005 \cdot (\,2\pi/n_{\mathrm{ref}}\,)$
Smoothing tolerance, $\delta\alpha_{\mathrm{min}}$	0.01
Target antenna lobe height	75 m
Target antenna beamwidth	60°
Target KOZ semi-axes	$[35 \;\; 50 \;\; 15]$ m

implementation details. First, for clarity we list the simulation parameters used in Table 1. Second, all position-related constraint-checking regard the chaser spacecraft as a point at its center of mass, with all other obstacles artificially inflated by the radius of its circumscribing sphere. Third and finally, all trajectory collision-checking is implemented by point-wise evaluation with a fixed time-step resolution Δt, using the analytic state transition equations Eq. (5) together with steering solutions from the The Steering Problem subsection of Section 3 to propagate graph edges; for speed, the line segments between points are excluded. Except near very sharp obstacle corners, this approximation is generally not a problem in practice (obstacles can always be inflated further to account for this). To improve performance, each obstacle primitive (ellipsoid, right-circular cone, hypercube, *etc.*) employs hierarchical collision-checking using hyper-spherical and/or hyper-rectangular bounding volumes to quickly prune points from consideration.

Motion Planning Solution

A representative solution to the posed planning scenario, both with and without the trajectory smoothing algorithm (Algorithm 2), is shown in Fig. 9. As shown, the planner successfully finds safe trajectories within each subplan, which are afterwards linked to form an overall solution. The state

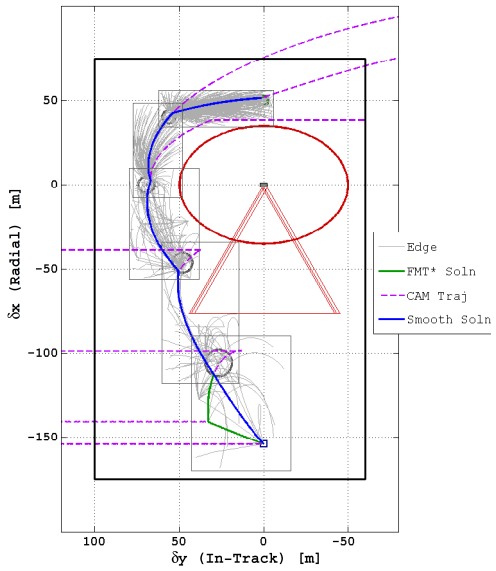
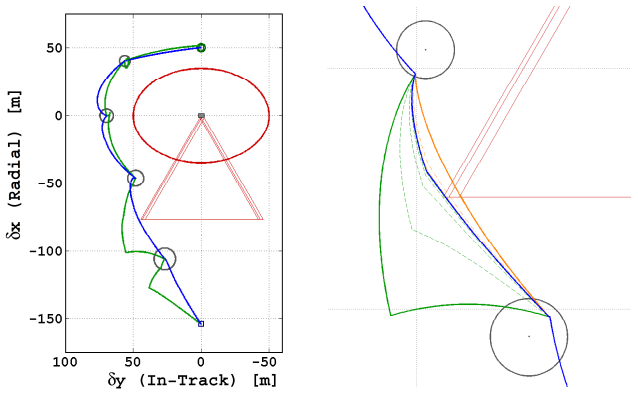


Figure 9. Representative planar motion planning solution using the FMT* algorithm (Algorithm 1) with $n = 2000$ (400 per subplan), $\bar{J} = 0.3$ m/s, and relaxed waypoint convergence. The output from FMT* is shown in green, while the trajectory combined with post-processing smoothing is shown in blue. Explored trajectories found to be safe are shown in grey. Actively-safe minimum-propellant abort trajectories are shown as purple dashed lines (one for each burn $\mathbf{\Delta v}_i$ along the trajectory).



(a) Paths before and after smoothing ($n = 2000$, $\bar{J} = 0.2$ m/s, exact waypoint convergence).

(b) Smoothing algorithm iterations ($n = 1500$, $\bar{J} = 0.3$ m/s, inexact waypoint convergence).

Figure 10. Visualizing trajectory smoothing (Algorithm 2) for the solution shown in Fig. 9, zoomed in on the second plan. The original plan is shown in green (towards the bottom-left), along with various iterates attempted while converging to the smoothed trajectory shown in blue (in the center). Invalid trajectories, including the lower propellant-cost trajectory used to guide the process, are shown in orange (towards the right).

space of the first subplan shown at the bottom is essentially obstacle-free, as the chaser at this point is too far away from the target for plume impingement to come into play. This means every edge connection attempted here is added; so the first subplan illustrates well a discrete subset of the reachable states around $\mathbf{x}_{\text{init}}$ and the unrestrained growth of FMT*. As the second subplan is reached, the effects of the Keep-Out-Zone position constraints come in to play, and we see edges begin to take more leftward loops. In subplans 3 and 4, plume impingement begins to play a role. Finally, in subplan 5 at the top, where it becomes very cheap to move between states (as the spacecraft can simply coast to the right for free), we see the initial state connecting to nearly every sample in the subspace, resulting in a straight shot to the final goal. As is evident, straight-line path planning would not approximate these trajectories well, particularly near coasting arcs which our dynamics allow the spacecraft to transfer to for free.

To understand the smoothing process, examine Fig. 10. Here we see how the discrete trajectory sequence from our sampling-based algorithm may be smoothly and continuously deformed towards the unconstrained minimal-propellant trajectory (as outlined in Section 4) until it meets trajectory constraints; if these constraints happen to be inactive, then the exact minimal-propellant trajectory is returned, as Fig. 10(a) shows. This computational approach is generally quite fast, assuming a well-implemented convex solver is used, as will be seen in the results of the next subsection.

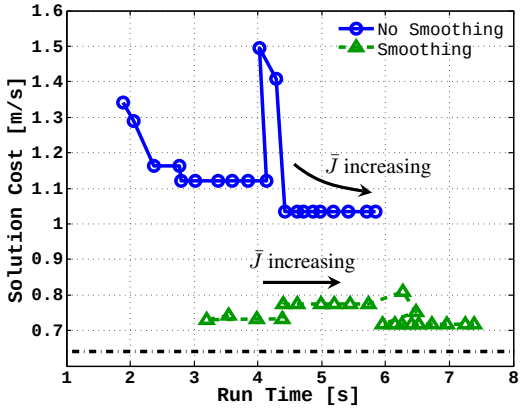
The 2-norm Δv costs of the two reported trajectories in this example come to 0.835 m/s (unsmoothed) and 0.811 m/s (smoothed). Compare this to 0.641 m/s, the cost of the unconstrained direct solution that intercepts each of the goal waypoints on its way to rendezvousing with $\mathbf{x}_{\text{goal}}$ (this trajectory exits the state-space along the positive in-track direction, a violation of our proposed mission; hence its cost represents an under-approximation to the true optimal cost J^* of the constrained problem). This suggests that our solutions are quite close to the constrained optimum, and certainly on the right order of magnitude. Particularly with the addition of smoothing at lower sample counts, the approach appears to be a viable one for spacecraft planning.

If we compare the 2-norm Δv costs to the actual measured propellant consumption given by the sum total of all allocated thruster Δv magnitudes, which equal 1.06 m/s (unsmoothed) and 1.01 m/s (smoothed) respectively, we find increases of 27.0% and 24.5%; as expected, our 2-norm cost metric under-approximates the true propellant cost. For point-masses with isotropic control authority (*e.g.*, a steerable or gimbaled thruster that is able to point freely in any direction), our cost metric would be exact. However, for our distributed attitude-dependent propulsion system (see Fig. 8(a)), it is clearly a reasonable proxy for allocated propellant use, returning values on the same order of magnitude. Though we cannot make a strong statement about our proximity to the propellant-optimal solution without directly optimizing over thruster Δv allocations, our solution clearly seems to promote low propellant consumption.

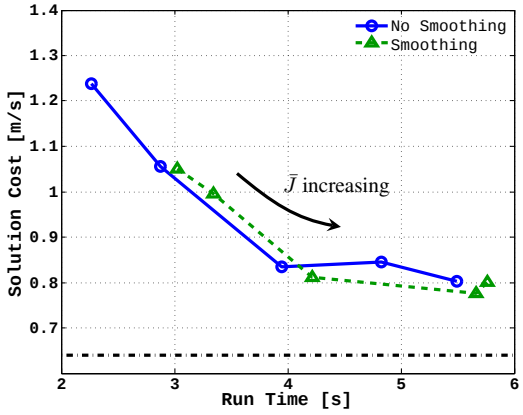
Performance Evaluation

To evaluate the performance of our approach, an assessment is necessary of solution quality as a function of planning parameters, most importantly the number of samples n taken and the reachability set cost threshold $\bar{J}$. As proven in the Theoretical Characterization of Section 3, the solution cost will eventually reduce to the optimal value as we increase the sample size n. Additionally, one can increase the cost threshold $\bar{J}$ used for nearest-neighbor identification so that more connections are explored. However, both come at the expense of running time. To understand the effects of these changes on quality, particularly at finite sample counts where the asymptotic guarantees of FMT* do not necessarily imply cost improvements, we measure the cost versus computation time for the planar planning scenario parameterized over several values of each n and $\bar{J}$.

Results are reported in Figs. 11–12. For a given sequence of sample count/cost threshold pairs, we ran our algorithm in each configuration and recorded the total cost of *successful*

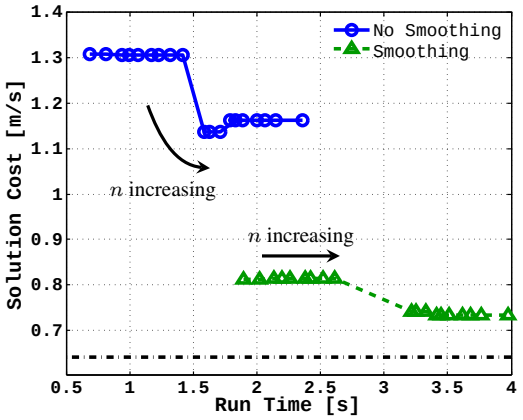


(a) Exact waypoint convergence ($n = 2000$).

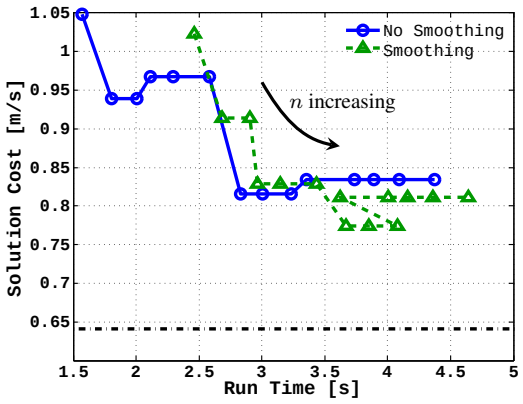


(b) Inexact waypoint convergence ($n = 2000$).

Figure 11. Algorithm performance for the given LEO proximity operations scenario as a function of varying cost threshold ($\bar{J} \in [0.2,\ 0.4]$) with n held constant (lowering $\bar{J}$ at these n yields failure). Results are reported for both (i) trajectories constrained to rendezvous exactly with pre-specified waypoints and (ii) trajectories that can terminate anywhere in $\mathcal{X}_{\text{goal}}$ (inside a given position/velocity tolerance).



(a) Exact waypoint convergence ($\bar{J} = 0.22$ m/s).



(b) Inexact waypoint convergence ($\bar{J} = 0.3$ m/s).

Figure 12. Algorithm performance for the given LEO proximity operations scenario as a function of varying sample count ($n \in [650,\ 2000]$) with $\bar{J}$ held constant (lowering n further at these $\bar{J}$ yields failure). Results are reported for trajectories both with and without exact waypoint convergence.

runs and their respective run times[3] as measured by wall clock time. Note that all samples were drawn and their interconnecting steering problems were solved *offline* per our discussion in Section 3. Only the *online* components of each call constitute the run times reported, including running FMT* with collision-checking and graph construction, as these are the only elements critical to the real-time implementability of the approach; everything else may be computed offline on ground computers where computation is less restricted, and later uplinked to the spacecraft or stored onboard prior to mission launch. See the Algorithm subsection of Section 3 for details. Samples were stored as a $d \times n$ array, while inter-sample steering controls $\mathbf{\Delta v}_i^*$ and times τ_i were precomputed as $n \times n$ arrays of $d/2 \times N$ and $N \times 1$ array elements, respectively. To reduce memory requirements, steering trajectories $\mathbf{x}^*$ and $\mathbf{q}$ were generated online through Eq. (5) and our nadir-pointing assumption, though in principle they could have easily been stored as well to save additional computation time.

Figure 11 reports the effects on the solution cost of varying the nearest-neighbor search threshold $\bar{J}$ while keeping n fixed. As described in the Reachability Sets subsection of Section 3, $\bar{J}$ determines the size of state reachability sets and hence the number of candidate neighbors evaluated during graph construction. Generally, this means an improvement in cost at the expense of extra processing; though there are exceptions as in Fig. 11(a) at $\bar{J} \approx 0.3$ m/s. Likely this arises from a neighbor that is found and connected to (at the expense of another, since FMT* only adds one edge per nearest-neighborhood) which leads to a particular graph change for which *exact* termination at the goal waypoint is more expensive than usual. Indeed we see that *for the same sample distribution* this does not occur, as shown in the other case where inexact convergence is permitted.

We can also vary the sample count n while holding $\bar{J}$ constant. From Figs. 11(a)–11(b), we select $\bar{J} = 0.22$ m/s and 0.3 m/s, respectively, for each of the two cases (the values which suggest the best solution cost per unit of run time). Repeating the simulation for varying sample count values, we obtain Fig. 12. Note the general downward trend as run time increases (corresponding to larger sample counts), indicating the classic trade-off in sampling-based planning. However, there is bumpiness. Similar to before, this is likely due to new connections previously unavailable at lower sample counts which cause a slightly different graph with an unlucky jump in propellant cost. This reinforces the well-known need to tune n and $\bar{J}$ before applying sampling-based planners.

[3]All simulations were implemented in MATLAB 2012b and run on a Windows-operated PC, clocked at 4.00 GHz and equipped with 32.0 GB of RAM. CVXGEN and CVX [37], disciplined convex programming solvers, were used to implement Δv allocation and trajectory smoothing, respectively.

As the figures show, the utility of trajectory smoothing is clearly affected by the fidelity of the planning simulation. In each, trajectory smoothing yields a much larger improvement in cost at modest increases in computation time when we require exact waypoint convergence. It provides little improvement, on the other hand, when we relax these waypoint tolerances; FMT* (with goal region sampling) seems to return trajectories with costs much closer to the optimum in such cases, making the additional overhead of smoothing less favorable. This conclusion is likely highly problem-dependent; these tools must always be tuned to the particular application.

Note that the overall run times for each simulation are on the order of 1-5 seconds, including smoothing. This clearly indicates that FMT* can return high quality solutions in real-time for spacecraft proximity operations. Though run on a computer currently unavailable to spacecraft, we hope that our examples serve as a reasonable proof-of-concept; we expect that with a more efficient coding language and implementation, our approach would be competitive on spacecraft hardware.

6. Conclusions

A technique has been presented for efficiently automating minimum-propellant guidance during near-circular orbit proximity operations, enabling the computation of near-optimal collision-free trajectories in real time (on the order of 1-5 seconds for our numerical examples). The approach allows our modified version of the FMT* sampling-based motion planning algorithm to approximate the solution to the minimal-propellant trajectory control problem Eq. (1) under impulsive Clohessy-Wiltshire-Hill (CWH) dynamics. The method begins by discretizing the feasible space of Eq. (1) through state space sampling in the CWH Local-Vertical Local-Horizontal (LVLH) frame. Next, state samples and their forward *reachability sets*, which we have shown comprise sets bounded by unions of ellipsoids taken over steering maneuver duration, are precomputed offline and stored onboard the spacecraft together with their pairwise steering solutions. Finally, the FMT* algorithm (with built-in trajectory smoothing) is called online to efficiently construct a tree of trajectories through the feasible state space towards a goal region, returning a solution that satisfies a broad range of trajectory constraints (*e.g.*, plume impingement, control allocation feasibility, obstacle avoidance, *etc.*) or else reporting failure. If desired, additional post-processing using the techniques outlined in Section 4 are employed to reduce solution propellant cost.

The key breakthrough of our solution for autonomous spacecraft guidance is its judicious distribution of computations; in essence, only what *must* be computed onboard, such as collision-checking and graph construction, is computed online – everything else, including the most intensive computations, are relegated to the ground where computational effort and run time are less critical. Furthermore, only minimal information (steering problem control trajectories, costs, and nearest-neighbor sets) requires storage on the spacecraft. Though we have illustrated through simulations the ability to tackle a particular minimum-propellant LEO homing maneuver problem, it should be noted that the methodology applies equally well to other objectives, such as the minimum-time problem, and can be generalized to other dynamic models and environments. The approach is flexible enough to handle non-convexity and mixed state-control-time constraints without compromising real-time implementability, so long as constraint function evaluation is relatively efficient. In short, the proposed approach appears to be useful for automating the mission planning process for spacecraft proximity operations, enabling real-time computation of low cost trajectories.

In future work, the authors plan to demonstrate the proposed approach in a number of other proximity operations scenarios, including deep-space guidance, pinpoint asteroid descent, and onboard a set of free-flying, air-bearing robots. However, the proposed planning framework for impulsively-actuated spacecraft offers several other interesting avenues for future research. For example, though nothing in the methodology forbids it outside of computational limitations, it would be interesting to revisit the problem with attitude states included in the planning process (instead of abstracted away, as we have done here by assuming an attitude profile). This would allow direct inclusion of attitude constraints into maneuver planning (*e.g.*, enforcing line-of-sight, keeping solar panels oriented towards the Sun to stay power positive, maintaining a communication link between the chaser antenna and ground, *etc.*). Extensions to dynamic obstacles (such as debris or maneuvering spacecraft, which are unfixed in the LVLH frame), elliptical target orbits, higher-order gravitation, curvilinear coordinates, or dynamics under relative orbital elements also represent key research topics vital to extending the method's applicability to more general maneuvers. Finally, memory and run time performance evaluations of our algorithms on space-like hardware would be necessary in assessing their true benefit to spacecraft planning in practice.

Acknowledgments

This work was supported by an Early Career Faculty grant from NASA's Space Technology Research Grants Program (Grant NNX12AQ43G).

References

[1] D. P. Dannemiller, "Multi-Maneuver Clohessy-Wiltshire Targeting," in *AAS Astrodynamics Specialist Conference*, Girdwood, AK, Jul. 2011, pp. 1–15.

[2] H. B. Hablani, M. L. Tapper, and D. J. Dana Bashian, "Guidance and Relative Navigation for Autonomous Rendezvous in a Circular Orbit," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 25, no. 3, pp. 553–562, 2002. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/2.4916

[3] B. J. Naasz, "Safety Ellipse Motion with Coarse Sun Angle Optimization," in *Proc. of the NASA GSFC Flight Mechanics Symposium*, Greenbelt, MD, Oct. 2005, pp. 1–13.

[4] D. E. Gaylor and B. W. Barbee, "Algorithms for Safe Spacecraft Proximity Operations," in *AAS Meeting*, ser. Advances in the Astronautical Sciences, vol. 127, no. 107, Seattle, WA, Jan. 2007, pp. 1–20.

[5] M. Develle, Y. Xu, K. Pham, and G. Chen, "Fast Relative Guidance Approach for Autonomous Rendezvous and Docking Control," in *Proc. of SPIE*, ser. Sensors and Systems for Space Applications IV, vol. 8044, Orlando, FL, apr 2011, pp. 80 440F.1–80 440F.15. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=1265645

[6] I. Lopez and C. R. McInnes, "Autonomous Rendezvous using Artificial Potential Function Guidance," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 18, no. 2, pp. 237–241, Mar. 1995. [Online]. Available:

http://arc.aiaa.org/doi/abs/10.2514/3.21375

[7] J. D. Muñoz and N. G. Fitz Coy, "Rapid Path-Planning Options for Autonomous Proximity Operations of Spacecraft," in *AAS Astrodynamics Specialist Conference*, Toronto, Canada, Aug. 2010, pp. 1–24. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/6.2010-7667

[8] M. W. Harris and B. Açıkmeşe, "Lossless Convexification of Non-Convex Optimal Control Problems for State Constrained Linear Systems," *Automatica*, vol. 50, no. 9, pp. 2304–2311, 2014.

[9] L. Breger and J. P. How, "Safe Trajectories for Autonomous Rendezvous of Spacecraft," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 31, no. 5, pp. 1478–1489, 2008.

[10] R. Vazquez, F. Gavilan, and E. F. Camacho, "Trajectory Planning for Spacecraft Rendezvous with On/Off Thrusters," in *IFAC World Congress*, vol. 18, no. 1, Milano, Italy, Aug. 2011, pp. 8473–8478.

[11] P. Lu and X. Liu, "Autonomous Trajectory Planning for Rendezvous and Proximity Operations by Conic Optimization," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 36, no. 2, pp. 375–389, Mar. 2013.

[12] Y.-Z. Luo, L.-B. Liang, H. Wang, and G.-J. Tang, "Quantitative Performance for Spacecraft Rendezvous Trajectory Safety," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 34, no. 4, pp. 1264–1269, Jul. 2011.

[13] A. Richards, T. Schouwenaars, J. P. How, and E. Feron, "Spacecraft Trajectory Planning With Avoidance Constraints Using Mixed-Integer Linear Programming," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, pp. 755–765, 2002.

[14] S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamic Planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

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

[16] J. M. Phillips, L. E. Kavraki, and N. Bedrossian, "Spacecraft Rendezvous and Docking with Real-Time, Randomized Optimization," in *AIAA Conf. on Guidance, Navigation and Control*, Austin, TX, Aug. 2003, pp. 1–11.

[17] M. Kobilarov and S. Pellegrino, "Trajectory Planning for CubeSat Short-Time-Scale Proximity Operations," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 37, no. 2, pp. 566–579, Mar. 2014. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.60289

[18] M. Haught and G. Duncan, "Modeling Common Cause Failures of Thrusters on ISS Visiting Vehicles," NASA, Tech. Rep., Jan. 2014, available at http://ntrs.nasa.gov/search.jsp?R=20140004797.

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

[20] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast Marching Tree: A Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions," *International Journal of Robotics Research*, vol. 34, no. 7, pp. 883–921, 2015.

[21] I. M. Ross, "How to Find Minimum-Fuel Controllers," in *AIAA Conf. on Guidance, Navigation and Control*, Providence, RI, Aug. 2004, pp. 1–10. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/6.2004-5346

[22] W. H. Clohessy and R. S. Wiltshire, "Terminal Guidance System for Satellite Rendezvous," *AIAA Journal of the Aerospace Sciences*, vol. 27, no. 9, pp. 653–658, Sep. 1960.

[23] G. W. Hill, "Researches in the Lunar Theory," *JSTOR American Journal of Mathematics*, vol. 1, no. 1, pp. 5–26, 1878.

[24] M. Bodson, "Evaluation of Optimization Methods for Control Allocation," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, pp. 703–711, Jul. 2002.

[25] G. Dettleff, "Plume Flow and Plume Impingement in Space Technology," *Progress in Aerospace Sciences*, vol. 28, no. 1, pp. 1–71, 1991.

[26] J. A. Starek, B. Açıkmeşe, I. A. D. Nesnas, and M. Pavone, "Spacecraft Autonomy Challenges for Next Generation Space Missions," in *Advances in Control System Technology for Aerospace Applications*, ser. Lecture Notes in Control and Information Sciences, E. Feron, Ed. Springer, Sep. 2015, vol. 460, ch. 1, pp. 1–48. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-662-47694-9_1

[27] J. A. Starek, B. W. Barbee, and M. Pavone, "A Sampling-Based Approach to Spacecraft Autonomous Maneuvering with Safety Specifications," in *AAS GN&C Conference*, ser. Advances in the Astronautical Sciences, vol. 154, Breckenridge, CO, Feb. 2015, pp. 1–13. [Online]. Available: http://www.univelt.com/book=4969

[28] K. Alfriend, S. R. Vadali, P. Gurfil, J. How, and L. Breger, *Spacecraft Formation Flying: Dynamics, Control and Navigation*. Butterworth-Heinemann, 2009, vol. 2.

[29] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[30] L. Janson, B. Ichter, and M. Pavone, "Deterministic Sampling-Based Motion Planning: Optimality, Complexity, and Performance," in *International Symposium on Robotics Research*, Sestri Levante, Italy, Sep. 2015, in Press.

[31] J. H. Halton, "On the Efficiency of Certain Quasirandom Sequences of Points in Evaluating Multidimensional Integrals," *Numerische Mathematik*, vol. 2, pp. 84–90, 1960.

[32] R. Allen and M. Pavone, "A Real-Time Framework for Kinodynamic Planning with Application to Quadrotor Obstacle Avoidance," in *AIAA Conf. on Guidance, Navigation and Control*, San Diego, CA, Jan. 2016, submitted.

[33] Joseph A. Starek, Edward Schmerling, Gabriel D. Maher, Brent W. Barbee, and Marco Pavone, "Fast, Safe, and Propellant-Efficient Spacecraft Planning under Clohessy-Wiltshire-Hill Dynamics," Jan. 2016, available at http://arxiv.org/abs/1601.00042.

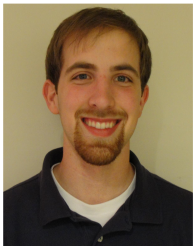
[34] E. Schmerling, L. Janson, and M. Pavone, "Optimal Sampling-Based Motion Planning under Differential Constraints: the Drift Case with Linear Affine Dynamics," in *Proc. IEEE Conf. on Decision and Control*, 2015, in Press.

[35] ——, "Optimal Sampling-Based Motion Planning under Differential Constraints: the Driftless Case," in *Proc. IEEE Conf. on Robotics and Automation*, 2015, pp. 2368–2375.

[36] S. N. Goward, J. G. Masek, D. L. Williams, J. R. Irons, and R. J. Thompson, "The Landsat 7 Mission: Terrestrial Research and Applications for the 21st Century," *Remote Sensing of Environment*, vol. 78, no. 1–2, pp. 3–12, 2001. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0034425701002620

[37] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming, Version 2.1," http://cvxr.com/cvx, Jun. 2015. [Online]. Available: http://cvxr.com/cvx/

BIOGRAPHY



Joseph A. Starek *received his B.S.E. and M.S.E. degrees in aerospace engineering from the University of Michigan at Ann Arbor in 2009 and 2010, respectively. He is currently a Ph.D. student in Aeronautics & Astronautics at Stanford University. His research interests include real-time spacecraft trajectory optimization, autonomous control algorithms, and actively-safe orbit design.*

Edward Schmerling *received a B.Sc. degree in mathematics and physics from Stanford University in 2010. He is currently working towards a Ph.D. degree at the Institute for Computational and Mathematical Engineering, Stanford University. His research interests include developing algorithms for robotic motion planning subject to general dynamic and cost constraints.*

Gabriel D. Maher *received his B.Sc. and M.Sc. degrees in aerospace engineering from the Delft University of Technology in The Netherlands in 2011 and 2014. He is now a Ph.D. student at the Institute for Computational and Mathematical Engineering at Stanford University. The current focus of his research is on the computational aspects of motion planning algorithms.*

Brent W. Barbee *is an Aerospace Engineer in the Navigation and Mission Design Branch of NASA's Goddard Space Flight Center; he also teaches graduate courses in the Department of Aerospace Engineering at The University of Maryland. Mr. Barbee currently supports flight dynamics for the OSIRIS-REx asteroid sample return mission launching in September 2016, is the technical lead for NASA's Near-Earth Object Human Space Flight Accessible Targets Study (NHATS), and conducts research on planetary defense against hazardous near-Earth objects. Mr. Barbee is a recipient of NASA's Early Career Achievement Medal and the AIAA National Capital Section 2012-2013 Hal Andrews Young Engineer/Scientist of the Year Award. He received his Bachelor and Master of Science degrees in Aerospace Engineering from the University of Texas at Austin.*

Dr. Marco Pavone *is an Assistant Professor of Aeronautics and Astronautics at Stanford University, where he is the Director of the Autonomous Systems Laboratory. Before joining Stanford, he was a Research Technologist within the Robotics Section at the NASA Jet Propulsion Laboratory. He received a Ph.D. degree in Aeronautics and Astronautics from the Massachusetts Institute of Technology in 2010. Dr. Pavone's areas of expertise lie in the fields of controls and robotics. His main research interests are in the development of methodologies for the analysis, design, and control of autonomous systems, with an emphasis on autonomous aerospace vehicles and large-scale robotic networks. He is a recipient of an NSF CAREER Award, a NASA Early Career Faculty Award, a Hellman Faculty Scholar Award, and was named NASA NIAC Fellow in 2011. He is currently serving as an Associate Editor for the IEEE Control Systems Magazine.*