A Hierarchical Control Architecture for Optimal Guidance and Navigation of Satellite Formations

Yasaman Pedari[1], Himadri Basu[2], Calum Buchanan[3], James Bagrow[4], Luis Duffaut Espinosa[1], Mads R. Almassalkhi[1], and Hamid R. Ossareh[1]

[1]*Department of Electrical and Biomedical Engineering, University of Vermont, Burlington, VT 05405, USA. Emails: {ypedari, lduffaut, malmassa, hossareh}@uvm.edu*

[2]*Department of Electrical and Computer Engineering, University of California, Santa Cruz, CA 95064, USA. Email: hbasu@ucsc.edu*

[3]*Department of Mathematics, University of Denver, Denver, CO 80210, USA. Email: calum.buchanan@du.edu*

[4]*Department of Mathematics and Statistics, University of Vermont, Burlington, VT 05405, USA. Email: jbagrow@uvm.edu*

This work considers an autonomous satellite swarm in low Earth orbit that must safely and efficiently reconfigure under uncertainties. We develop a control architecture that assigns a desired terminal formation and autonomously drives the swarm from its initial configuration to that formation within a prescribed time, while meeting requirements on scalability, safety, fuel efficiency, and resilience to communication failures. The proposed hierarchical approach separates decision-making across time scales and exploits swarm redundancy to improve communication-network reliability when available satellites exceed mission-required positions. The architecture comprises a planning layer and a control layer. The planning layer includes (*i*) a network-reliability-maximizing formation planner and (*ii*) a fuel-optimal, collision-aware trajectory planner. The control layer includes (*i*) a real-time trajectory-tracking controller with obstacle avoidance and (*ii*) a robust estimator that fuses global positioning system measurements and inter-satellite ranging to maintain accurate state estimates in the presence of sensor outliers. The framework assumes identical satellites and axis-decoupled translational actuation, with attitude stabilized by a separate controller. The architecture is validated through high-fidelity nonlinear simulations, and hardware-in-the-loop testing of the trajectory planner demonstrates computational feasibility on flight-class hardware.

Nomenclature

General Notations

$\|.\|_p$	:	l_p Norm of a vector, $p \in [1, \infty)$
$[\cdot]_i$	:	Operator that returns the i-th column when applied to a matrix or the i-th element when applied to a vector
$\lfloor \cdot \rfloor$	:	Floor operator
I_p	:	Identity matrix of size $p \times p$
$0_p, 0_{p\times q}$	:	$p \times 1$ Zero vector and $p \times q$ zero matrix
$\text{diag}(\cdot)$	:	Operator that maps a vector to a square diagonal matrix with the vector entries on the diagonal
$\mathcal{N}(\mu, \sigma)$	:	Normal distribution with mean μ and standard deviation σ
N	:	Number of satellites within the formation
S_i	:	The ith satellite within the formation
R_{det}	:	Detection radius of the inter-satellite distance measurement sensor
R_{comm}	:	Distance within which a pair of satellites can communicate
R_{safe}	:	Minimum allowed inter-satellite distance to avoid collision risk
t_f	:	Terminal time of the maneuver
k	:	Sampling step size
K	:	Terminal time-step
T_c	:	Coarse discretization step-size of the trajectory planner
T_f	:	Fine discretization step-size of the trajectory planner
T_r	:	Discretization step-size of the real-time controller
T_e	:	Discretization step-size of the real-time estimator
$r_i(k)$	:	Relative position of satellite i at time-step k in the LVLH frame
$v_i(k)$	:	Relative velocity of satellite i at time-step k in the LVLH frame
$x_i(k)$	:	Relative state of satellite i at time-step k in the LVLH frame, defined as $x_i(k) = \text{col}\big(r_i(k), v_i(k)\big)$
$u_i(k)$	:	Control input vector for satellite i at time-step k in the LVLH frame
$\mathbf{x}_{\text{des}}$	:	Desired (terminal) relative state matrix of the swarm in the LVLH frame, size $6 \times N$
$\mathbf{x}_{\text{ini}}$	:	Initial relative state matrix of the swarm in the LVLH frame, size $6 \times N$
$x_i^*(k)$	:	Optimal trajectory for the i^{th} satellite calculated by the trajectory planner
$u_i^*(k)$	:	Optimal control vector for the i^{th} satellite calculated by the trajectory planner
$\hat{r}_i(k)$	:	Estimated relative position of satellite i at time-step k in the LVLH frame
$\hat{v}_i(k)$	:	Estimated relative velocity of satellite i at time-step k in the LVLH frame
$\hat{x}_i(k)$	:	Estimated relative state of satellite i at time-step k in the LVLH frame, defined as $\hat{x}_i(k) = \text{col}\big(\hat{r}_i(k), \hat{v}_i(k)\big)$

$y_i(k)$	:	Vector of position measurements for the i^{th} satellite at time step k (obtained from both GPS or inter-satellite ranging)

Subscripts and Superscripts

ini	=	Initial condition ($t = 0$)
des	=	Desired final condition ($t = t_f$)

I. Introduction

A. Motivation

Future space missions aim to observe dynamic events and changing environments in low Earth orbit with a level of spatial and temporal coverage that cannot be achieved by any single spacecraft. Long-term NASA and ESA visions describe distributed space systems that operate as a single, highly coordinated virtual instrument, capable of autonomously reconfiguring to mission objectives [1, 2]. For example, [3] envisions coordinated multi-perspective observations of the heliosphere. Beyond enabling measurement modalities that exceed a single spacecraft, multiple satellites can offer greater resilience and versatility than a single monolithic one. They can also deliver higher mission capability at a fraction of the cost (i.e., in distributed imaging, atmospheric sampling, and in-space assembly of large structures [4–6]). As a case in point, consider two Earth-imaging missions: Landsat 8 (one large satellite; \$855 million) and Planet Labs' Dove constellation (hundreds of small satellites; \$120 million); both pursue the same imaging goal, but the latter achieved higher-frequency imaging at a fraction of the cost [7, 8].

In recent years, there has been a surge in formation flights, showing both the promise and the remaining gaps of multi-satellite missions. Table 1 summarizes more than twenty flown missions from 2000–2025 [9, 10]. In the table, the missions are classified by (*i*) autonomy level, where *highly autonomous* refers to onboard Guidance, Navigation, and Control (GNC) with only supervisory ground oversight, whereas *partial-to-high ground-controlled* denotes limited onboard decision-making, and (*ii*) inter-satellite separation, where *close proximity* indicates a mission in which the inter-satellite separations are around or less than 1 km, and *loose formation* refers to separations of ∼1–500 km. Note that Table 1 is not exhaustive and is simply used to identify trends rather than to provide a census. Thus, three patterns emerge: (1) close-proximity missions have been demonstrated mainly with two-spacecraft pairs; (2) for loose formations, teams of up to four spacecraft have flown, but most missions remain ground-controlled or only partially autonomous; and (3) across all cases, demonstrations with more than four spacecraft have not been widely reported.

Taken together, these patterns point to a capability gap: autonomy at scale (i.e., tens or hundreds of satellites), especially below 1 km. Nevertheless, the benefits of addressing this gap are significant (e.g., improved mission resilience and flexibility at reduced cost). To address this gap, we consider a formation of small satellites in low Earth orbit (LEO) with a mission-driven objective (e.g., coverage/observation performance [11] or communication quality [12]). We

Table 1 Multi-satellite missions, 2000–2025, categorized by autonomy and separation.

	Partial to High Ground-controlled	**Highly autonomous**
Close-proximity formation	• AeroCube-10 (2019, N=2) • OCSD (AeroCube-7B/7C) (2017, N=2) • CanX-4/CanX-5 (2014, N=2) • TerraSAR-X/TanDEM-X (2007/2010, N=2) • EO-1/Landsat-7 (2000, N=2)	• Proba-3 (2024, N=2) • CPOD (2022, N=2) • PRISMA (Mango/Tango) (2010, N=2) • NASA DART/MUBLCOM (2005, N=2) • Orbital Express (2007, N=2) • BIROS/BEESAT-4 (2016, N=2)
Loose formation	• UNSW M2 A/B (2021, N=2) • GRACE–FO (2018, N=2) • MMS (2015, N=4) • ESA Swarm (2013, N=3) • ELISA (2011, N=4) • ESSAIM (2004, N=4) • GRACE (2002, N=2) • Cluster II (2000, N=4)	• NASA Starling (2023, N=4) • V-R3x (2021, N=3) • Adelis–SAMSON (2021, N=3)

assume that the formation includes more satellites than are required to meet this mission objective, leaving redundancy in the swarm, and we exploit this redundancy to improve reliability. This setup is consistent with large-swarm and constellation design architectures [13–15]. For example, in Planet Labs' PlanetScope [16] and Iridium/Iridium NEXT constellations [17], coverage requirements are met with fewer prescribed roles than available satellites, leaving excess satellites to provide robustness. The robustness metric that we optimize for in this paper is the reliability of the inter-satellite communication network, which enables distributed computations and sensor fusion across the formation.

Under this setup, the goal of our proposed architecture, at a high level, is to assign a desired terminal formation for the swarm, then to guide and navigate the swarm from its initial state to the desired formation within a prescribed time. All computations are expected to be carried out autonomously onboard the swarm rather than on the ground, and the framework must satisfy the following requirements: (*i*) **Scalability:** the framework must be able to manage tens to hundreds of satellites; (*ii*) **Fuel efficiency:** it must ensure fuel-efficient maneuvers that respect fuel budgets; (*iii*) **Safety:** it must guarantee robust, collision-free navigation in the presence of modeling uncertainty, disturbances, and noisy measurements; and (*iv*) **Reliability:** it must maintain mission objectives in the presence of hardware failures, inter-satellite-link (ISL) outages, and sensing dropouts. A high-level diagram of the proposed framework is shown in Fig. 1. Details will be provided in Section I.C.

We make the following hardware and modeling assumptions about the satellites: each satellite is equipped with GPS sensors, inter-satellite communication and ranging, and axis-decoupled translational actuation provided by a lower-level controller. An attitude controller is assumed to decouple attitude from the translational dynamics, so attitude dynamics are neglected. All satellites are treated as identical, and all inter-satellite links are subject to failures from multiple sources (sensing outages, communication dropouts, or temporary blockage), which are captured via a probabilistic link-failure model.

B. Literature Review

End-to-end, unified GNC frameworks for satellite swarms are rare in the literature (see, e.g., [18–20]); however, there is substantial work on their constituent components: (*i*) Formation planning & reconfiguration, (*ii*) Collision-aware guidance & control, and (*iii*) Relative navigation & state estimation. In what follows, we review the literature within each component and identify gaps in light of our mission context and requirements:

Formation Planning & Reconfiguration: Formation planning refers to designing the terminal swarm configuration to achieve mission objectives. Prior work assumed the final swarm geometry largely as a given, often set by mission design or simple robustness rules without fully exploiting redundancy in the number of agents, see, e.g., [21–24]. In particular, references [23, 24] take a network robustness perspective that models the swarm as a communication graph and reasons about connectivity under node and link failures. From a graph-theoretic standpoint, network robustness is often quantified using surrogates such as algebraic connectivity. However, these surrogates do not directly quantify connectivity under probabilistic node and link failures. All-terminal reliability does, as the probability the graph remains connected, but it is computationally intractable to evaluate exactly at scale [25].

Collision-Aware Guidance & Control: There exist a plethora of multi-agent motion planning and control methodologies, including optimal-control/MPC approaches [26, 27], evolutionary or genetic search [28], local-policy methods (e.g., artificial potential fields, equilibrium shaping, behavior-based, consensus) [29–31], and leader–follower architectures [32]. Nevertheless, close-proximity swarms with limited onboard computation/communications, tight propellant budgets, and large swarm sizes have limitations. For instance, centralized optimal control approaches suffer from poor scaling in both computation and communication, open-loop policies lack safety guarantees, local policies can be propellant-inefficient and can fall into undesired equilibria, and leader–follower methods induce single-point failures and weak feedback on formation quality [33, 34]. These considerations motivate a hierarchical decomposition that separates global trajectory assignment from fast and constraint-aware tracking.

Relative Navigation & State Estimation: In current formation-flying missions, relative position and velocity are most commonly estimated using extended and unscented Kalman filters, which propagate a relative dynamic model and update it using available positioning measurements [35, 36]. The underlying process models typically range from simplified linearized representations to higher-fidelity ones. In flight, such Kalman-filter-based estimators have been deployed on missions including PRISMA [35], CanX-4/5 [37], and TanDEM-X [38], where GPS measurements, sometimes augmented by inter-satellite ranging or other onboard sensors, are fused in real time to provide the relative state needed for guidance and control. Almost all high-accuracy formation-flying systems rely on some form of GPS-based relative measurement within these filters.

As a result, as these estimators are mainly driven by GPS-based measurements, any limitations of GPS directly translate into limitations in navigation performance. For example, the availability and quality of GPS can degrade during ionospheric activity, under poor satellite geometry (e.g., high latitudes), and due to multipath propagation and

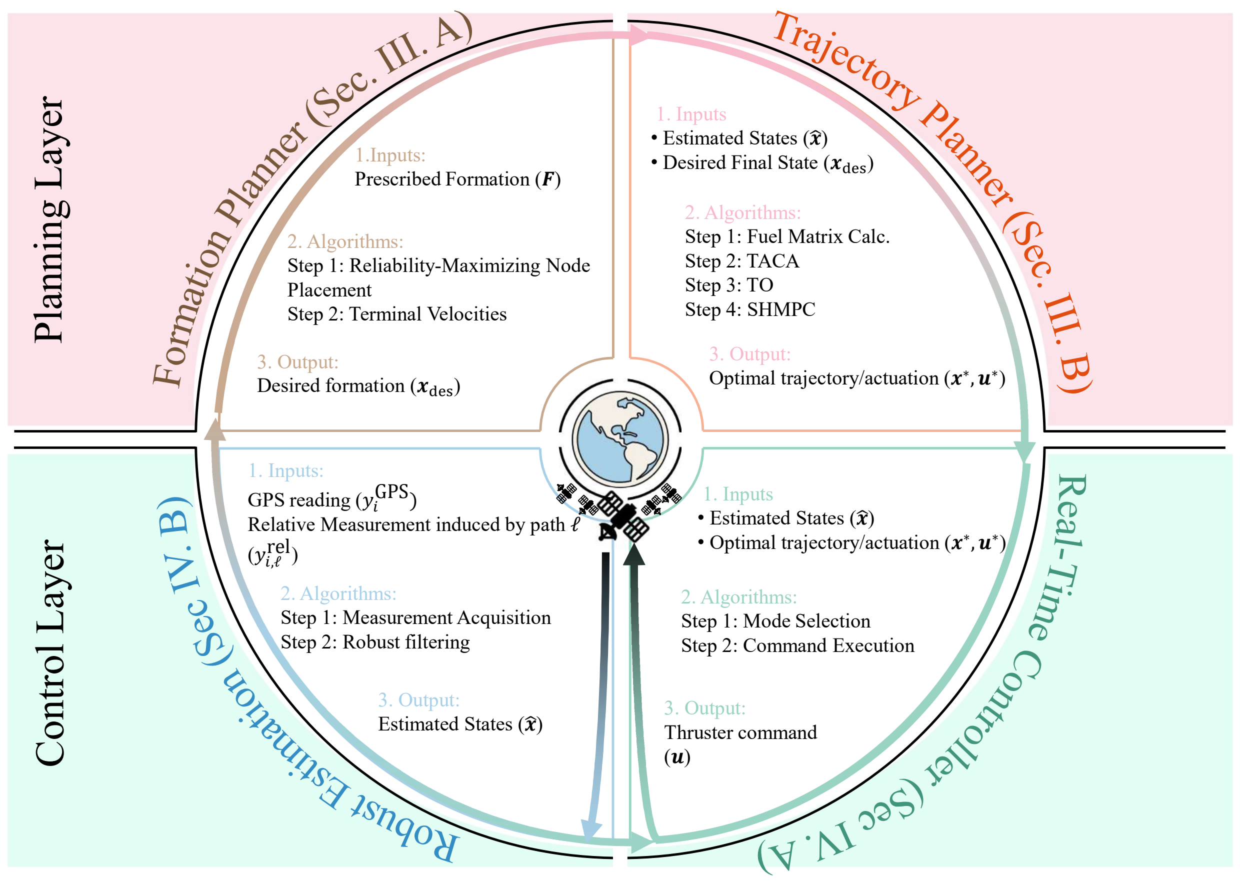

Planning Layer
Control Layer
Formation Planner (Sec. III. A)
Trajectory Planner (Sec. III. B)
Robust Estimation (Sec IV. B)
Real-Time Controller (Sec IV. A)
1.Inputs:
Prescribed Formation ($\boldsymbol{F}$)
2. Algorithms:
Step 1: Reliability-Maximizing Node Placement
Step 2: Terminal Velocities
3. Output:
Desired formation ($\boldsymbol{x}_{\text{des}}$)
1. Inputs
• Estimated States ($\hat{\boldsymbol{x}}$)
• Desired Final State ($\boldsymbol{x}_{\text{des}}$)
2. Algorithms:
Step 1: Fuel Matrix Calc.
Step 2: TACA
Step 3: TO
Step 4: SHMPC
3. Output:
Optimal trajectory/actuation ($\boldsymbol{x}^*, \boldsymbol{u}^*$)
1. Inputs:
GPS reading (y_i^{GPS})
Relative Measurement induced by path ℓ ($y_{i,\ell}^{\text{rel}}$)
2. Algorithms:
Step 1: Measurement Acquisition
Step 2: Robust filtering
3. Output:
Estimated States ($\hat{\boldsymbol{x}}$)
1. Inputs
• Estimated States ($\hat{\boldsymbol{x}}$)
• Optimal trajectory/actuation ($\boldsymbol{x}^*, \boldsymbol{u}^*$)
2. Algorithms:
Step 1: Mode Selection
Step 2: Command Execution
3. Output:
Thruster command ($\boldsymbol{u}$)

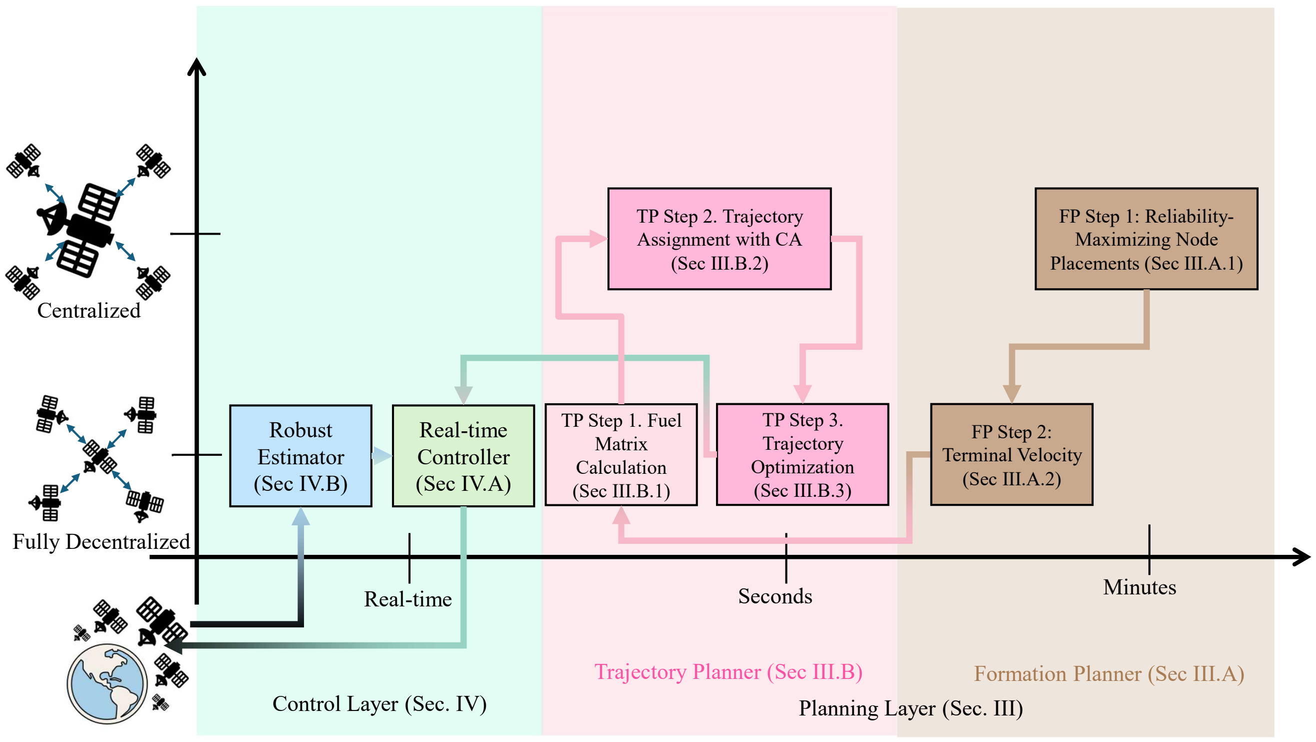
Fig. 1 Hierarchical Architecture of the GNC framework. The framework is divided into a Planning Layer (Formation and Trajectory Planning) and a Control Layer (Robust Estimation and Real-Time Control)

reflections [39–41]. When GPS is degraded or denied (e.g., jamming/spoofing), these Kalman filter-based estimators can accumulate large errors [42]. Robust, outlier-tolerant estimators, e.g., [43], offer a viable path to maintain navigation quality in such conditions, yet they remain under-deployed in flight systems. While these navigation challenges are not unique to swarms, close-proximity formations provide abundant inter-satellite relative measurements that enable cooperative estimation and maintain high-accuracy relative state estimates even when GPS degrades [44].

C. Proposed Framework and Original Contributions

The requirements introduced in Section I.A necessitate decision making across multiple time-scales: from microsecond safety reactions to hour-level formation redesign. Such operation is inherently multi-time-scale, and no centralized controller can reason effectively across this bandwidth, especially when each satellite has partial knowledge of the formation and must cope with noise, uncertainty, hardware failures, and obstacles in real time. These constraints make a using a centralized controller for the entire swarm computationally infeasible.

To address this gap, we propose an architecture that separates computation-intensive global decision making from fast, safety-critical execution. Accordingly, we adopt a hierarchical two-layer architecture, shown in Fig. 1. In the figure, the upper half shows the outer layer, i.e., the Planning Layer (PL). The PL performs computation-intensive algorithms and includes the formation planner and the trajectory planner. The formation planner in the top-left receives



TP Step 2. Trajectory Assignment with CA (Sec III.B.2)
FP Step 1: Reliability-Maximizing Node Placements (Sec III.A.1)
Centralized
Robust Estimator (Sec IV.B)
Real-time Controller (Sec IV.A)
TP Step 1. Fuel Matrix Calculation (Sec III.B.1)
TP Step 3. Trajectory Optimization (Sec III.B.3)
FP Step 2: Terminal Velocity (Sec III.A.2)
Fully Decentralized
Real-time
Seconds
Minutes
Trajectory Planner (Sec III.B)
Formation Planner (Sec III.A)
Control Layer (Sec. IV)
Planning Layer (Sec. III)

Fig. 2 Comparison of the GNC framework components' computation frequency and implementation strategy. Implementation types include: Fully Decentralized (on-board/individual) and Centralized (single satellite or ground computation).

a prescribed mission-defined formation that is underspecified because the swarm contains redundant satellites, then leverages this redundancy to select a terminal formation that maximizes reliability against inter-satellite-link failures (Section III.A). The resulting terminal states are provided to the trajectory planner in the top-right, which generates time-parameterized, collision-free, fuel-efficient trajectories that respect thrust limits, keep-out zones, and timing requirements (Section III.B). These trajectories are then sent to the inner layer.

The lower half of Fig. 1 shows the inner layer, i.e., the Control Layer (CL). The controller in the bottom-right computes actuator commands at millisecond time-scales to track the planned trajectories while ensuring collision avoidance (Section IV.A). The estimator in the bottom-left fuses GPS measurements using outlier-tolerant methods to maintain accurate relative state estimates under GPS degradation and sensor anomalies (Section IV.B). These estimates are provided both to the PL trajectory planner and to the real-time controller.

Figure 2 describes the two layers in terms of computation time and whether the computations are centralized or decentralized. The horizontal axis shows the computation time for each component of the proposed architecture, from real-time computations in the Control Layer to second-level computations in the Trajectory Planner and minute-level computations in the Formation Planner. The vertical axis indicates whether each component is implemented in a fully decentralized manner on board individual satellites (bottom) or in a centralized manner on a single satellite or even on the ground (top).

Note that our requirements impose competing objectives in terms of reliability and fuel efficiency. The proposed framework targets reliability through the formation planner and fuel efficiency through the trajectory planner, treating these objectives in a deliberately decoupled manner. This separation preserves computational tractability in the planning layer at the expense of global optimality. Solving both objectives jointly would lead to a large, nonconvex optimization problem that is impractical for flight hardware with tight timing constraints [45, 46]. However, simulation results (Section V) show that mission-level reliability remains high even without explicitly enforcing reliability in the trajectory planner, and that the resulting trajectories achieve fuel usage comparable to precision formation-flying data.

The contributions of this manuscript can then be summarized as follows. **First,** we develop a hierarchical, scalable swarm GNC architecture that integrates (*i*) a novel connectivity-preserving formation planner based on all-terminal reliability, extending our prior work in [47] (from 2D, single-node position placement in a fixed-geometry formation to 3D assignment of the full formation, including velocity), (*ii*) our group's previously developed fuel- and computationally efficient trajectory planner [48], and (*iii*) a novel low-level control layer that couples real-time safety-critical control with robust state estimation, integrating and extending [49, 50] for scalable formations, into a single end-to-end system. Thus, the key novelties of the first contribution are extension of our prior work in [47] and the integration of the various components into a unified framework for swarm GNC. **Second,** we present high-fidelity simulation validation of the GNC framework in a CanX-4/5-inspired scenario, benchmarking performance against reported mission results. **Third,** we perform hardware-in-the-loop (HIL) testing to assess the computational feasibility of the trajectory planner for onboard execution.

The rest of the manuscript is organized as follows. Section II develops the relative-dynamics models used across the framework and for high-fidelity benchmarking. Section III details the planning layer: the formation planner (Sec. III.A) and the trajectory planner (Sec. III.B). Section IV covers the control layer: the real-time controller (Sec. IV.A) and the robust estimator (Sec. IV.B). Finally, Section V validates the integrated framework via high-fidelity simulations and hardware-in-the-loop tests.

II. Satellite Relative Dynamics Modeling

We use two models of satellite relative motion. The first is a high-fidelity nonlinear model from [51] that includes the dominant LEO perturbations (J_2 and atmospheric drag). The second is a linearized J_2 model from [52] that captures J_2 while neglecting air drag. The nonlinear model underpins the formation-flight simulations in Section V. Although well suited for high-fidelity simulation, using it for predictive tasks such as trajectory planning yields non-convex problems that become computationally intractable for large swarms, with no practical guarantees of global optimality. A linear model is therefore required for tractable optimization and efficient planning at scale.

Classical linear models often rely on assumptions that limit accuracy, e.g., circular reference orbits (Clohessy–

Wiltshire [53], Schweighart–Sedwick [54]), neglected oblateness and drag, and validity only for small inter-satellite separations, thus, a model is needed that is linear yet sufficiently accurate for predictive purposes. As shown in [55], the linearized J_2 model achieves trajectory-prediction errors within 10 m over up to 1.3 orbital periods when the swarm remains within 1.5 km of the target orbit. Building on these results, we use the linearized J_2 model for predictive tasks in trajectory optimization and estimation, and, where necessary, in control. We first present the nonlinear model, then the linearized J_2 model.

A. High-fidelity Nonlinear Model

Satellite relative motion is commonly described in the Local Vertical Local Horizontal (LVLH) frame, a rotating frame centered on a (real or virtual) reference satellite, whose trajectory defines the *target orbit*. In the LVLH frame, one axis points from Earth's center to the reference satellite (radial outward), a second axis is perpendicular to the orbital plane and points opposite the orbit normal, and the third axis lies in the orbital plane and points forward along the reference satellite's velocity, forming a right-handed frame. Each satellite's motion in the formation is then defined in the LVLH frame, i.e., relative to the reference satellite.

The target-orbit dynamics are characterized by six parameters: radial distance ρ, radial velocity v_x, angular momentum h, Right Ascension of the Ascending Node Ω, inclination i, and argument of latitude θ. Together, these uniquely define the orbit and its motion. The equations governing the orbital dynamics are expressed as follows:

$$\dot{\rho} = v_x \tag{1a}$$

$$\dot{v}_x = -\frac{\mu}{\rho^2} + \frac{h^2}{\rho^3} - \frac{k_{J_2}}{\rho^4}\left(1 - 3\sin^2 i \sin^2\theta\right) - C\|\vec{V}_a\| v_x \tag{1b}$$

$$\dot{h} = -\frac{k_{J_2}\sin^2 i \sin 2\theta}{\rho^3} - C\|\vec{V}_a\|\left(h - \omega_e\rho^2\cos i\right) \tag{1c}$$

$$\dot{\Omega} = -\frac{2k_{J_2}\cos i \sin^2\theta}{h\rho^3} - \frac{C\|\vec{V}_a\|\omega_e\rho^2\sin 2\theta}{2h} \tag{1d}$$

$$\dot{i} = \frac{k_{J_2}\sin 2i \sin 2\theta}{2h\rho^3} - \frac{C\|\vec{V}_a\|\omega_e\rho^2\sin i\cos^2\theta}{h} \tag{1e}$$

$$\dot{\theta} = \frac{h}{\rho^2} + \frac{2k_{J_2}\cos^2 i\sin^2\theta}{h\rho^3} + \frac{C\|\vec{V}_a\|\omega_e\rho^2\cos i\sin 2\theta}{2h}, \tag{1f}$$

where $J_2 = 1.08263 \times 10^{-3}$, $k_{J_2} = \frac{3}{2}J_2\mu R_e^2$, $R_e \approx 6378\,\text{km}$, $\mu = GM \approx 3.986 \times 10^{14}\,\text{m}^3\,\text{s}^{-2}$ (with $G = 6.67430 \times 10^{-11}\,\text{m}^3\,\text{kg}^{-1}\,\text{s}^{-2}$ and $M = 5.972 \times 10^{24}\,\text{kg}$), C denotes the air-drag coefficient (shape/surface dependent), $\|\vec{V}_a\|$ is the atmosphere-relative speed, and $\omega_e \approx 7.2921 \times 10^{-5}\,\text{rad}\,\text{s}^{-1}$.

After identifying the target orbit, the relative motion of the i-th satellite within the formation, with respect to this target orbit, is described by the following dynamics, where $r_i \in \mathbb{R}^3$ denotes its position in the LVLH frame, v_i its corresponding velocity, and together they define the satellite state $x_i = \mathrm{col}(r_i, v_i)$:

$$\dot{r}_i = v_i, \tag{2a}$$

$$\dot{v}_i = \begin{bmatrix} 2[v_i]_2\,\omega_z - [r_i]_1(\eta_i^2 - \omega_z^2) + [r_i]_2\,\alpha_z - [r_i]_3\,\dot{\omega}_x\omega_z - (\zeta_i - \zeta)\sin i \sin\theta - \rho(\eta_i^2 - \eta^2) \\ -C_i\|\vec{V}_i\|\big([v_i]_1 - [r_i]_2\omega_z\big) - \big(C_i\|\vec{V}_i\| - C\|\vec{V}_a\|\big)\,v_x \\ \\ -2[v_i]_1\,\omega_z + 2[v_i]_3\,\omega_x - [r_i]_1\,\alpha_z - [r_i]_2(\eta_i^2 - \omega_z^2 - \omega_x^2) + [r_i]_3\,\alpha_x - (\zeta_i - \zeta)\sin i\cos\theta \\ -C_i\|\vec{V}_i\|\big([v_i]_2 + [r_i]_1\omega_z - [r_i]_3\omega_x\big) - \big(C_i\|\vec{V}_i\| - C\|\vec{V}_a\|\big)\left(\frac{h}{\rho} - \omega_e\rho\cos i\right) \\ \\ -2[v_i]_2\,\omega_x - [r_i]_1\,\omega_x\omega_z - [r_i]_2\,\alpha_x - [r_i]_3(\eta_i^2 - \omega_x^2) - (\zeta_i - \zeta)\cos i \\ -C_i\|\vec{V}_i\|\big([v_i]_3 + [r_i]_2\omega_x\big) - \big(C_i\|\vec{V}_i\| - C\|\vec{V}_a\|\big)\,\omega_e\rho\cos\theta\cos i \end{bmatrix} + u_i. \tag{2b}$$

where

$$\begin{aligned}
\zeta &= \frac{2k_{J2}\sin i\sin\theta}{\rho^4}, & \zeta_i &= \frac{2k_{J2}\rho_{iZ}}{\rho^5},\\
\eta^2 &= \frac{\mu}{\rho^3} + \frac{k_{J2}}{\rho^5} - \frac{5k_{J2}\sin^2 i\sin^2\theta}{\rho^5}, & \eta_i^2 &= \frac{\mu}{\rho_i^3} + \frac{k_{J2}}{\rho_i^5} - \frac{5k_{J2}\rho_{iZ}^2}{\rho_i^5},\\
\rho_i &= \sqrt{(\rho + [r_i]_1)^2 + [r_i]_2^2 + [r_i]_3^2}, & \rho_{iZ} &= (\rho + [r_i]_1)\sin i\sin\theta + [r_i]_2\sin i\cos\theta + [r_i]_3\cos i,\\
\omega_x &= \dot{i}\cos\theta + \dot{\Omega}\sin\theta\sin i, & \omega_z &= \dot{\theta} + \dot{\Omega}\cos i,\\
\alpha_x &= \dot{\omega}_x, \quad \alpha_z = \dot{\omega}_z
\end{aligned}$$

Here C_i is the follower's drag coefficient. The dynamics can be written compactly as

$$\dot{x}_i = \mathrm{col}\big(v_i,\ \mathcal{G}(r_i, v_i, t)\big) + \mathrm{col}(0,\ u_i). \tag{3}$$

B. J_2 Linearized Model

Following [52], the nonlinear terms n_j^2 and ζ_j in (3) (polynomials in the reciprocals of the components of r_i) admit a Gegenbauer expansion that is linear in r_i. Applying this linearization and neglecting air drag yields the first-order linear time-varying J_2 model.

$$\dot{x}_i = \begin{bmatrix} \multicolumn{3}{c}{0_{3\times3}} & \multicolumn{3}{c}{I_3} \\ 2\eta^2 + \omega_z^2 + \frac{2kJ_2}{\rho^5}\left(1 - \sin^2 i\,\sin^2\theta\right) & \alpha_z + \frac{4kJ_2\sin^2 i\,\sin 2\theta}{\rho^5} & -5\,\omega_x\omega_z & 0 & 2\omega_z & 0 \\ \frac{4kJ_2\sin^2 i\,\sin 2\theta}{\rho^5} - \alpha_z & -\left(\frac{2kJ_2\sin^2 i\,\cos^2\theta}{\rho^5} + \eta^2 - \omega_x^2 - \omega_z^2\right) & \alpha_x - \frac{kJ_2\sin 2i\,\cos\theta}{\rho^5} & -2\omega_z & 0 & 2\omega_x \\ -5\,\omega_x\omega_z & -\left(\frac{kJ_2\sin 2i\,\cos\theta}{\rho^5} + \alpha_x\right) & -\left(\eta^2 - \omega_x^2 + \frac{2kJ_2\cos^2 i}{\rho^5}\right) & 0 & -2\omega_x & 0 \end{bmatrix} x_i + \begin{bmatrix} 0_{3\times3} \\ I_3 \end{bmatrix} u_i \tag{4}$$

The above matrix is time-varying since its coefficients are functions of the orbital parameters $\rho(t)$, $i(t)$, and $\theta(t)$; in particular, ω_x, ω_z, η, α_x, and α_z vary with these parameters. For implementation, we discretize the above model:

$$x_i(k+1) = A(k)\, x_i(k) + B\, u_i(k). \tag{5}$$

The discretization step size is chosen depending on the subproblem under study. Specifically, the trajectory planner and the estimator use this model at different timescales: the coarse and fine trajectory planner updates occur with periods T_c and T_f, respectively, while the estimator runs with period T_e. Whenever (5) is invoked, the specific sampling period will be stated explicitly, as it determines the corresponding discrete matrices $A(k)$ and B obtained from Euler discretization of the continuous-time dynamics in (4).

III. Planning Layer

The planning layer governs the swarm's high-level decisions by selecting the desired formation and generating the trajectories to achieve it. As shown in Fig. 1, the planning layer forms the outer layer of the hierarchical architecture and operates on slower time scales than the control layer. It comprises (*i*) a formation planner that exploits swarm redundancy to select a terminal formation with high communication network reliability and (*ii*) a trajectory planner that computes fuel-efficient trajectories to reach that formation; together, these components map mission-level formation specifications to optimal trajectories that the control layer must safely track in real time.

A. Formation Planning

In this section, we address the problem of planning satellite formations with robust communication networks. By robust, we mean that the communication network is likely to remain connected in the presence of uncertainties which may render some ISLs inoperable (e.g., due to hardware failures, radiation, atmospheric effects, or obstructions) [56–59]. Recalling the formation redundancy assumption, we assume that the desired terminal geometry is specified only for M satellites, with $M < N$ (where N denotes the size of the formation), and is given by $\mathbf{F}_0 \in \mathbb{R}^{3\times M}$. The formation planner exploits this underspecification of the terminal state to compute a full desired terminal formation for the swarm while optimizing network reliability.

As ISL failures occur randomly, we model each ISL as independently operational with probability $p \in [0, 1]$, and failing with probability $1 - p$. In this work, we quantify the robustness of the network using its *all-terminal reliability* defined as the probability that the graph remains connected after each edge fails independently with probability $1 - p$ [25]. Note that maximizing all-terminal reliability in a formation by placing the remaining $N - M$ satellites can be cast as an optimization problem. This problem is nonconvex and NP-hard, making global optimality difficult to achieve for large-scale swarms [60]. Rather than assigning positions to all $N - M$ redundant satellites at once, we adopt a greedy strategy that places them sequentially, each time selecting the placement that maximizes the all-terminal reliability, as in [47][*]. While this approach does not guarantee global optimality, our simulations show that it achieves near-optimal reliability with a small loss relative to an exhaustive randomized search, and at a fraction of the computational cost.

The swarm's communication network is modeled as an undirected graph whose vertices represent satellites and whose edges represent ISLs; such graphs are commonly referred to as unit ball graphs, generalizing unit disk graphs used in 2-D settings. Initialize the graph with the M prescribed satellites. Let $\mathbf{F}_0 \in \mathbb{R}^{3\times M}$ be the matrix whose jth column $[\mathbf{F}_0]_j \in \mathbb{R}^3$ is the LVLH position of satellite S_j, and define the normalized position map $f_0(j) = [\mathbf{F}_0]_j / R_{\text{comm}}$ for $j \in \{1, \dots, M\}$. We model the communication network by the undirected graph $G_0 = (V_0, E_0)$ with $V_0 = \{1, \dots, M\}$ and $E_0 = \{\{i, j\} \subseteq V_0 : \|f_0(i) - f_0(j)\|_2 \leq 1\}$. We assume that G_0 is a connected graph, that is, that the prescribed formation geometry $\mathbf{F}_0$ is such that any two satellites are joined by a path of inter-satellite links. Recall that the iterative algorithm has $N - M$ iterations. For any iteration k, let $f_k : V_k \to \mathbb{R}^3$ assign the normalized LVLH positions, and define edges by the unit ball rule $E_k = \big\{\{i, j\} \subseteq V_k : \ \|f_k(i) - f_k(j)\|_2 \leq 1\big\}$. That is, an edge is included whenever the normalized LVLH distance between satellites i and j is at most one. The planner assigns the remaining satellites by adding one node per iteration: for $k = 1, \dots, N - M$, add satellite $M + k$, choose its position $f_k(M + k)$, and update

$$G_k = (V_k, E_k) \qquad V_k = \{1, \dots, M + k\}.$$

After the final iteration, all N satellites are assigned and the terminal formation is $\mathbf{F}_{\text{des}} = R_{\text{comm}} \begin{bmatrix} f_{N-M}(1) & \cdots & f_{N-M}(N) \end{bmatrix}$. Note that here k denotes the graph-growth iteration leading to the terminal formation, whereas throughout the rest of the paper k is reserved for the discrete-time sampling index. Throughout this section, we assume the satellites in G_0 are in *general position* with respect to the normalized position map f_0, in particular that *(i)* $\|f_0(i) - f_0(j)\|_2 \neq 2$ for all distinct $i, j \in V_0$, and *(ii)* no four points $\{f_0(i)\}_{i \in V_0}$ lie on the boundary of a radius-1 sphere.

Finally, after the determination of $\mathbf{F}_{\text{des}}$, the planner must additionally determine terminal velocities $\mathbf{v}_{\text{des}} \in \mathbb{R}^{3\times N}$ for the formation, chosen so that all satellites share the same specific orbital energy to minimizes along-track drift, helping the formation preserve relative geometry after reconfiguration [61]. We summarize these design requirements on the output $\mathbf{x}_{\text{des}}$ of the formation planner, where $\mathbf{x}_{\text{des}} = \text{col}(\mathbf{F}_{\text{des}}, \mathbf{v}_{\text{des}})$, in the following problem statement.

Problem Statement 1.

Given: *Prescribed formation geometry* $\mathbf{F}_0$*, communication range* R_{comm}*, and link reliability* p*.*

Find: *A terminal formation state* $\mathbf{x}_{\text{des}} = \text{col}(\mathbf{F}_{\text{des}}, \mathbf{v}_{\text{des}})$ *such that:*

(i) The terminal geometry $\mathbf{F}_{\text{des}}$ *preserves the* M *ground-prescribed positions in* $\mathbf{F}_0$ *and assigns the remaining* $N - M$ *satellites to positions that iteratively maximize all-terminal reliability of the resulting communication*

[*]In [47], we provide an algorithm to implement this strategy in 2-D and analyze is effectiveness via simulations. Here, we implement the 3-D version.

network.

(ii) $\mathbf{v}_{\text{des}}$ *is determined to ensure orbital energy matching across all satellites.*

With the problem formally defined, we next outline how the proposed formation planner addresses it at a high level:

Step 0: **Activate** the formation planner under one of the following conditions: (*i*) a new prescribed formation geometry, $\mathbf{F}_0$, becomes available through higher-level mission planning; (*ii*) either satellite failures (requiring removal of the satellite and its ISLs) or individual ISL failures occur, necessitating changes in the desired formation design.

Step 1: **Iteratively** add the remaining $N - M$ satellites to maximize all-terminal reliability. For each iteration,

1.1. Enumerate all *feasible neighborhoods* that the new node can have in G_{k-1}; that is, all subsets $U \subseteq V_{k-1}$ of satellites in G_{k-1} that are contained in a radius-1 sphere which contains no points in $V_{k-1} \setminus U$.

1.2. For each candidate neighborhood U, estimate the reliability of the graph obtained by adding a vertex with neighborhood U to G_{k-1}. The neighborhood that yields the maximum reliability, $U_{\max}$, is selected.

1.3. Compute the optimal position $c^* \in \mathbb{R}^3$ as the center of the smallest enclosing sphere for $U_{\max}$. Update the formation to G_k, where the $(M + k)$-th node is inserted at c^* and is appended to G_{k-1}.

Step 2: **Assign** terminal velocities $\mathbf{v}_{\text{des}}$ consistent with $\mathbf{F}_{\text{des}}$:

2.1. Compute $\mathbf{v}_{\text{des}}$ such that all satellites have the same orbital energy. The output of the formation planning block is then $\mathbf{x}_{\text{des}} = \text{col}(\mathbf{F}_{\text{des}}, \mathbf{v}_{\text{des}}) \in \mathbb{R}^{6 \times N}$.

Regarding the estimation of all-terminal reliability in Step 1.2, we note that computing this value, even for the unit ball graphs considered here is $\sharp\mathsf{P}$-complete [62]. Therefore, in this work, whenever we require this value, we approximate it via Monte Carlo simulations. This method works by repeatedly sampling probabilistic graphs obtained from the original graph by deleting each edge independently with probability $1 - p$. A particularly efficient Monte Carlo scheme for this task is described in [63]. We now describe each step in detail.

1. Iterative Reliability-Maximizing Placement of Redundant Satellites (Step 1)

This step determines where to place the $(M + k)$-th satellite within the current graph G_{k-1} so as to maximize the all-terminal reliability of the updated graph. The procedure is carried out iteratively for $k = 1, \ldots, N - M$, thereby placing the redundant satellites from node $M + 1$ through node N. We now detail the three-step procedure (Steps 1.1–1.3) introduced earlier that is used to compute each placement.

We present an algorithm that enumerates the set $\mathcal{U}$ of all *feasible neighborhoods* for the $(M+k)$-th node (denoted by v in this section) to be added to G_{k-1}, i.e., all subsets of V_{k-1} that could be joined to v by an edge in G_k. The

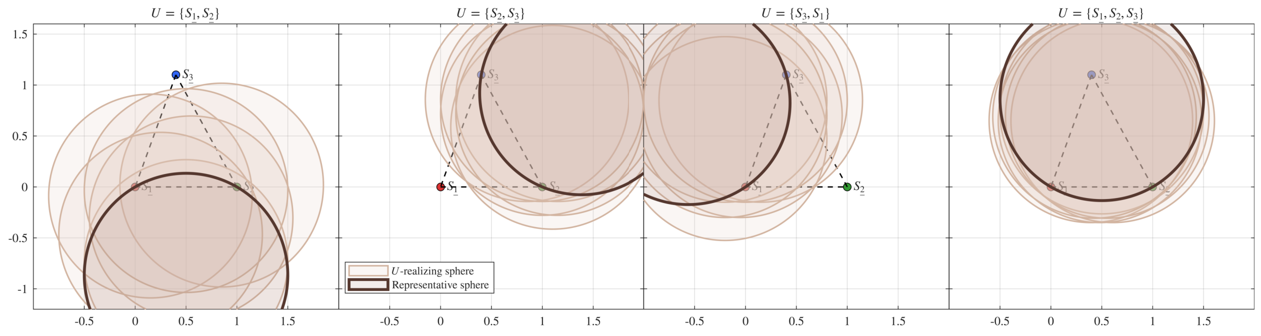


Fig. 3 For a fully-connected graph with three nodes, each subfigure shows multiple unit circles centered at feasible positions of v that realize the same neighborhood U and one representative sphere with two vertices of U on its boundary, per Prop. 1.

neighborhood of v, with position c, in G_{k-1} is the set of vertices $u \in V_{k-1}$ such that $\|f_{k-1}(u) - c\|_2 \leq 1$. Hence, to enumerate $\mathcal{U}$, we must enumerate all subsets $U \subseteq V_{k-1}$ that can be enclosed by a unit ball containing every vertex in U while excluding all vertices in $V_{k-1} \setminus U$, ensuring that U is the exact neighborhood of the added vertex. We call a unit ball S, *U-realizing* if *(i)* $f_{k-1}(j)$ lies in the interior of S for every $j \in U$, and (*ii*) every point in $V_{k-1} \setminus U$ lies outside S. The family of U-realizing spheres is typically infinite: translating or rotating any U-realizing sphere preserves this property until a vertex of V_{k-1} meets or leaves the boundary of S. Thus, neighborhood changes occur only when the boundary passes through a vertex. An illustration of this point is shown in Fig. 3, where a fully connected graph of three nodes is depicted. Each subfigure corresponds to a different neighborhood U. For each neighborhood, multiple faded circles illustrate multiple U-realizing spheres that realize the same neighborhood and lose this property only at boundary events. The following proposition asserts the existence of what we refer to as a *representative sphere* for the (infinite) family of U-realizing spheres associated with the same neighborhood, namely, a sphere that has at least two points from V_{k-1} on its boundary and contains all other points in U (but none in $V_{k-1} \setminus U$) in its interior.

Proposition 1. *Let V_{k-1} be a set of at least two points in $\mathbb{R}^3$, and let $U \subseteq V_{k-1}$ contain at least two points. If there exists a U-realizing radius-1 sphere that contains U in its interior but no points in $V_{k-1} \setminus U$, then there also exists a radius-1 sphere with two points $S_1, S_2 \in V_{k-1}$ on its boundary, and having precisely $U \setminus \{S_1, S_2\}$ as the set of points from V_{k-1} in its interior.*

Proof. Let S be a radius-1 sphere containing every point in U but no points in $V_{k-1} \setminus U$ in its interior. If some point from V_{k-1} lies on the boundary of S, call it S_1. Otherwise, let S_1 denote the point in V_{k-1} closest in distance to the boundary of S, and translate S in the direction of S_1 until S_1 lies on its boundary. Next, we rotate this translation of S around the point S_1 on its boundary (in any direction) until a second point S_2 lies on its boundary. We are sure to find such a point S_2 since $|U| \geq 2$. □

By the representative-sphere result in the preceding proposition, any feasible neighborhood can be represented by a unit ball whose boundary contains exactly two vertices, say S_1 and S_2. Starting from this representative sphere, if we rotate while keeping S_1 and S_2 on its boundary (i.e., rotations about the line $\overline{S_1S_2}$), then the realized neighborhood remains unchanged under this motion until the boundary first meets a third vertex S_3. At such a three-point boundary contact, let A denote the set of all non-boundary vertices currently in the interior of S, and let $B \subseteq \{S_1, S_2, S_3\}$ be any subset of the three boundary vertices (Recall that we assume that no four points lie on the boundary of a unit ball and that no two of S_1, S_2, and S_3 are of normalized distance exactly 2 apart). In the following proposition, we show that, under a general-position assumption, arbitrarily small admissible motions of the representative sphere can toggle each of S_1, S_2, and S_3 independently across the boundary, without including or excluding any other points from V_{k-1}. To further clarify, Fig. 4 shows a fully connected graph of 10 nodes. The same unit ball with three points exactly on its boundary is shown in all subfigures. In each subfigure, an additional sphere is shown to illustrate how toggling the original sphere enables realization of all possible B subsets, i.e., finding a sphere that arbitrarily includes or excludes boundary points. The following proposition formalizes this observation.

Proposition 2. *Let S_1, S_2, and S_3 be distinct points in $\mathbb{R}^3$ which are pairwise of distance strictly less than 2 apart and which lie on the boundary of a radius-1 sphere S. If A is a finite set of non-boundary points in S and $B \subseteq \{S_1, S_2, S_3\}$, then there is a radius-1 sphere containing $A \cup B$ but not containing $\{S_1, S_2, S_3\} \setminus B$.*

Proof. Note that $\{S_1, S_2, S_3\}$ contains no two antipodal points on S since the distances between them are strictly less than 2. Consider the line segment L with endpoints S_1 and S_2. By rotating S about L, we can either include or exclude S_3 from the interior of a radius-1 sphere S' with S_1 and S_2 on its boundary. If exactly one of S_1 or S_2 is in B, we shift S' in the direction of the vector from S_1 to S_2 to include S_2 but not S_1, or in the direction of the vector from S_2 to S_1 to include S_1 but not S_2. If both S_1 and S_2 are in B, we shift S' in the direction of the vector from its center through the midpoint of L. If neither S_1 nor S_2 is in B, we shift the sphere S' in the opposite direction. □

Propositions 1 and 2 are then utilized in the following theorem, which specifies the enumeration of feasible neighborhoods for a newly added node.

Theorem 1. *Let $V_{k-1} \subset \mathbb{R}^3$ be a finite set of at least three points in general position. A subset $U \subseteq V_{k-1}$ with $|U| \geq 3$ is a feasible neighborhood for the node v to be added if and only if $U \in \mathcal{U}$, where*

$$\mathcal{U} = \left\{ A \cup B \;\middle|\; \begin{array}{l} \exists \text{ distinct } S_1, S_2, S_3 \in V_{k-1} \text{ and a unit ball } S \subset \mathbb{R}^3 \text{ with } S_1, S_2, S_3 \in \partial S, \\ A = \left(S \cap V_{k-1}\right) \setminus \{S_1, S_2, S_3\},\ B \subseteq \{S_1, S_2, S_3\} \end{array} \right\}.$$

Here, ∂S denotes the boundary of the sphere S.

Proof. The proof is omitted for conciseness; it follows directly from Propositions 1 and 2. □

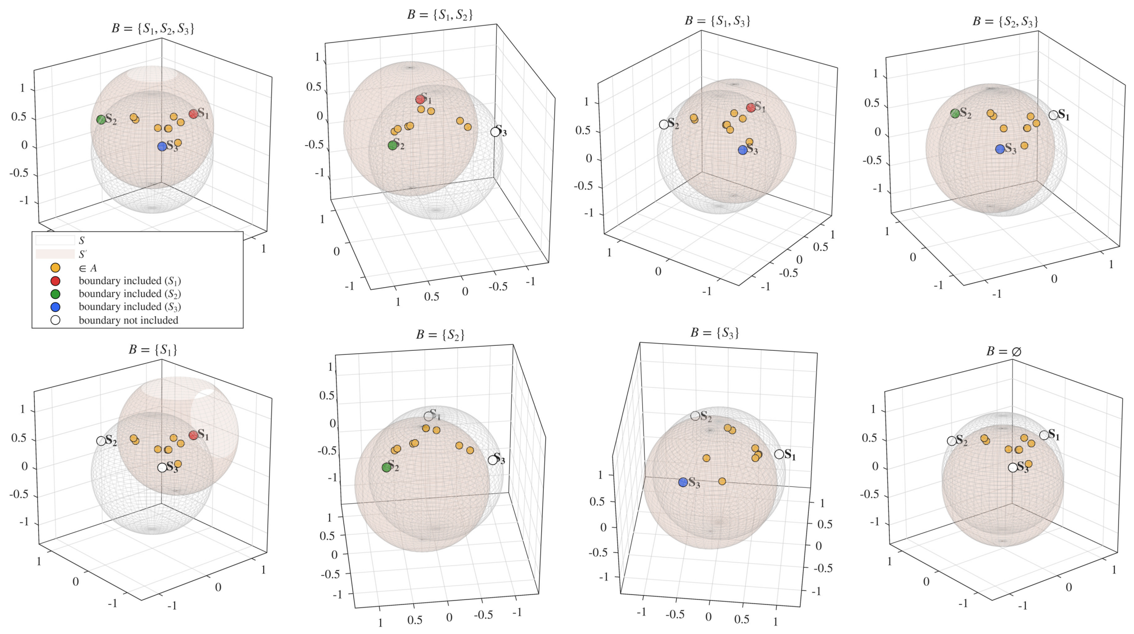


Fig. 4 Satellites S_1, S_2, S_3 lie on the boundary of the unit ball S. Gold points denote the interior set A. Each panel corresponds to a choice of $B \subseteq \{S_1, S_2, S_3\}$ and shows a sphere S' with $A \cup B \subset \mathrm{int}(S')$; filled markers indicate $\mathrm{S}_i \in B$, and hollow markers indicate exclusion.

We now present an algorithm, based on the theorem above, to enumerate all feasible neighborhoods for the to-be-added node v. Algorithm 1 proceeds as follows. For each unordered pair $\{S_1, S_2\} \subseteq V_{k-1}$ with $\|S_1 - S_2\|_2 < 2$, construct a unit ball S with $S_1, S_2 \in \partial S$, and let $A := \big(S \cap (V_{k-1})\big) \setminus \{S_1, S_2\}$ denote the set of interior vertices. By Proposition 2, initialize the neighborhood set as $\mathcal{U} \leftarrow \{A \cup B \mid B \subseteq \{S_1, S_2\}\}$. The ball S is then rotated about the line $\overline{S_1 S_2}$ (allowing infinitesimal radius-preserving translations), and the realized neighborhood is recorded until a boundary event occurs in which a third vertex S_3 meets ∂S. At such an event, a toggle operation is used to enumerate all sets of the form $A \cup B$ with $B \subseteq \{S_1, S_2, S_3\}$. Points near $\overline{S_1 S_2}$ (those swept by rotating the shortest boundary arc from S_1 to S_2) never exit the ball, while all other points both enter and exit exactly once per full rotation. Entry and exit angles follow from elementary trigonometry (cf. [47] for the two-dimensional case); for simplicity, explicit formulas are omitted. In practice, the rotation is discretized into l steps of size $2\pi/l$. During this sweep, whenever a vertex enters (resp. exits) the ball, it is inserted into (resp. removed from) the current interior set, and the resulting interior set and its unions with $\{S_1, S_2\}$ are appended to $\mathcal{U}$. After one full rotation, all feasible interior sets for the pair $\{S_1, S_2\}$ are recorded. Repeating this procedure over all pairs with $\|S_1 - S_2\|_2 < 2$, and invoking Proposition 1, yields all feasible maximal[†] neighborhoods. Duplicate neighborhoods are removed automatically since $\mathcal{U}$ is maintained as a set.

Equivalently, one can think of Algorithm 1 enumerating the regions of intersection cut out by the unit balls centered at the points in V_{k-1}. While there are exponentially many possible neighborhoods for a vertex in an abstract graph, it is

[†]The algorithm would not, for instance, find a feasible neighborhood consisting of a single satellite surrounded by a number of others at distance 1.1. However, such a neighborhood would not be maximal, and thus would not be a candidate for $U_{\max}$.

```
Algorithm 1: Finding subsets contained in a sphere
Result: Set $\mathcal{U}$ of sets of points contained in a radius-1 sphere
Input: Finite set $V_{k-1}$ of at least three points in $\mathbb{R}^3$, rotation discretization step $l$
Initialize $\mathcal{U} = \{\}$
for $S_1 \neq S_2 \in V_{k-1}$ with $\|S_1 - S_2\|_2 < 2$ do
    $S$ = arbitrary radius-1 sphere with $S_1, S_2$ on boundary, $\theta = 0$, $U = \{u \in V_{k-1} : u \in$ interior of $S\}$
    Add $U$, $U \cup \{S_1\}$, $U \cup \{S_2\}$, and $U \cup \{S_1, S_2\}$ to $\mathcal{U}$
    do
        Rotate $S$ around the line segment $\overline{S_1 S_2}$ according to $\theta$ until a third point $S_3 \in V_{k-1}$ is on the boundary
        if $S_3 \in U$ then
            Delete $S_3$ from $U$
            Add $U$, $U \cup \{S_1\}$, $U \cup \{S_2\}$, and $U \cup \{S_1, S_2\}$ to $\mathcal{U}$
        end
        if $S_3 \in V \setminus U$ then
            Add $S_3$ to $U$
            Add $U$, $U \cup \{S_1\}$, $U \cup \{S_2\}$, and $U \cup \{S_2, S_3\}$ to $\mathcal{U}$
        end
        $\theta \leftarrow \theta + \frac{2\pi}{l}$
    while $\theta < 2\pi$;
end
return $\mathcal{U}$
```

shown in [64] that N spheres partition $\mathbb{R}^3$ into at most $(N^3 - 3N^2 + 8N)/3$ regions. The arrangement bound implies $O(N^3)$ distinct feasible neighborhoods, which matches the runtime of our enumeration. Indeed, we perform $O(N^2)$ unit-sphere sweeps (one per boundary pair $\{S_i, S_j\}$), and each sweep has at most $O(N)$ contact events, yielding $O(N^3)$ time (up to polylogarithmic factors).

Having obtained the set of all possible neighborhoods $\mathcal{U}$ for a new vertex via Algorithm 1 (Step 1.1), we now determine the optimal neighborhood $U_{\max} \in \mathcal{U}$ that maximizes the overall graph reliability upon insertion of vertex v. Given the positions of the fixed vertices $\{f_k(1), \ldots, f_k(M + k - 1)\} \subset \mathbb{R}^3$, we formulate this as an optimization problem where the decision variable is the choice of neighborhood $U \in \mathcal{U}$, i.e.,

$$U_{\max} \in \arg\max_{U \in \mathcal{U}} \operatorname{Rel}\left((G_{k-1})^{+v}(U)\right),$$

where $(G_{k-1})^{+v}(U)$ is the graph obtained by adding node v to the graph G_{k-1} and connecting v exactly to the vertices in U.

Having found the neighborhood $U_{\max}$ for v that provides the most reliable graph (Step 1.2), we turn to assigning a particular location to this vertex (Step 1.3). In general, the optimal location may depend on the specific mission for the satellite swarm, but a natural candidate is the point that minimizes the maximum distance to a vertex in $U_{\max}$. This point is precisely the center of the smallest enclosing sphere for the desired neighborhood. One can use Welzl's randomized algorithm [65] to find this center in linear time. We take the optimal position for $U_{\max}$ to be c^* and update the vertex location accordingly by setting $f_k(v) \leftarrow c^*$.

2. *Assignment of Terminal Velocities (Step 2)*

So far, the formation planner has provided the desired formation $G_{N-M} = (V_{N-M}, E_{N-M})$ with embedding $f_{N-M} : V_{N-M} \to \mathbb{R}^3$; the unnormalized terminal positions of the formation in the chief's LVLH frame are $\mathbf{F}_{\text{des}} = \text{col}(f_{N-M}(1), \dots, f_{N-M}(N))R_{\text{comm}}$, and the next step is to assign the corresponding velocity vectors $\mathbf{v}_{\text{des}}$. We assign the velocities to ensure the entire formation evolves with the same mean motion and preserves the designed configuration over time. Following [61], the velocities are adjusted so that every satellite has the same specific orbital energy, guaranteeing identical mean motion and preventing secular drift of the formation. A closed-form velocity assignment satisfying this condition is given in [61] and is used in our work. Finally, the full formation state $\mathbf{x}_{\text{des}} = \text{col}(\mathbf{F}_{\text{des}}, \mathbf{v}_{\text{des}})$ is the output of the formation planner and an input to the trajectory planner (next section), determining the terminal state of the formation.

In summary, this section developed a computationally tractable formation planner that, given desired positions for only a subset of satellites, autonomously assigns a terminal formation for the entire swarm that maximizes network reliability. Key innovations that enable this are (*i*) explicitly leveraging redundancy by using the remaining satellites to maximize network reliability, thereby reducing dependence on the ground and allowing the swarm to adapt autonomously to failures, (*ii*) adopting all-terminal reliability as the network reliability metric, which is rarely used in satellite formation planning and accurately captures the probability of staying connected under probabilistic link and node failures than common graph surrogates, (*iii*) solving the problem iteratively by placing the redundant satellites one by one rather than solving a single coupled optimization for all of them at once, which keeps the computation manageable while giving up little in terms of all-terminal reliability, (*iv*) introducing graph-based abstractions that enumerate feasible neighborhoods in the induced unit ball graph, reducing the continuous search over three-dimensional positions to a finite set of candidates that can be evaluated efficiently, and (*v*) assigning terminal velocities so that all satellites share the same specific orbital energy (and hence similar orbital periods), which helps preserve the designed communication network over time.

B. Trajectory Planning (TP)

In this section, we provide an overview of our trajectory planner, originally introduced in [48] as TP2 and included here for the sake of completeness. The trajectory planner, given the current (initial) state $\mathbf{x}_{\text{ini}} \in \mathbb{R}^{6\times N}$ of the formation as provided by the estimator in the control layer, is responsible for planning trajectories and optimal actuation sequences x_i^* and u_i^* that are fuel-optimal and safe, and that take the formation to the terminal formation planned by the FP, $\mathbf{x}_{\text{des}}$, over a predetermined duration t_f. We summarize the design requirements in the following problem statement.

Problem Statement 2.

Given: *The initial swarm state* $\mathbf{x}_{\text{ini}}$*; desired terminal state* $\mathbf{x}_{\text{des}}$*; terminal time* t_f*; planner fine sampling period* T_f*; planner coarse sampling period* T_c*; safety radius* R_{safe}*; actuator limit* $u_{\max}$*; and slew-rate bounds* $u_{r,\min}$, $u_{r,\max}$.

Find: *For each* $i \in \{1, \ldots, N\}$*, an optimal trajectory* $\mathbf{x}_i^*(k)$ *and actuation sequence* $\mathbf{u}_i^*(k)$ *over* $k = 0, \ldots, K$*, where* $K = \lfloor t_f / T_f \rfloor$*, such that:*

(i) ***Safety:*** $\|\mathbf{r}_i^*(k) - \mathbf{r}_j^*(k)\|_2 \geq R_{\text{safe}}$ *for all* $i \neq j$ *and all* k.

(ii) ***Boundary conditions:*** $\mathbf{x}_i^*(0) = [\mathbf{x}_{\text{ini}}]_i$ *and* $\mathbf{x}_i^*(K) = [\mathbf{x}_{\text{des}}]_i$.

(iii) ***Fuel optimality:*** *The total fuel proxy is minimized,* $\min \sum_{i=1}^{N} \sum_{k=0}^{K-1} \|\mathbf{u}_i^*(k)\|_1$.

(iv) ***Robustness:*** *Robustness to modeling uncertainties and perturbations (as defined in Section 2) is maintained.*

(v) ***Actuation limits:*** *actuation satisfies* $\|\mathbf{u}_i^*(k)\|_\infty \leq u_{\max}$*, and* $u_{r,\min} \leq \mathbf{u}_i^*(k+1) - \mathbf{u}_i^*(k) \leq u_{r,\max}, \quad \forall i,\ k$.

(vi) ***Computational scalability:*** *The optimization solves within* 1 *minute of computation for* $N > 10$ *satellites.*

Our TP addresses this problem through 4 steps:

Step 0: **Activate** the trajectory planner under one of the following: (*i*) a new terminal formation $\mathbf{x}_{\text{des}}$ becomes available from the formation planner; (*ii*) the Shrinking Horizon Model Predictive Control (SHMPC) logic (defined in this section) triggers a timer, upon which the TP recomputes optimal trajectories for robustness to uncertainties.

Step 1: **Fuel Matrix calculation**: the cost of each satellite going to each destination is evaluated.

Step 2: **Trajectory Assignment with Collision Avoidance (TACA):** assign which satellite targets which destination, subject to collision-avoidance constraints.

Step 3: **Trajectory Optimization (TO):** given the assignment, each satellite finds the most fuel-optimal path to its destination.

Step 4: **Shrinking Horizon Model-Predictive Control (SHMPC):** a predictive controller whose prediction horizon initial recedes until it reaches a terminal condition and then the planning horizon shrinks. The control schedules and resulting predicted trajectories are regularly re-optimized to adapt to model uncertainties and disturbances.

In what follows, we present each step one by one.

1. Fuel matrix calculation (Step 1)

In step 1, the satellites compute, in a fully decentralized way (*i.e.*, using local computing resources), the minimum fuel required to go from their current location as given by the estimator $\mathbf{x}_{\text{ini}}$ to each of the locations in the final configuration, $\mathbf{x}_{\text{des}}$. This fuel estimate does not include collision-avoidance (CA) constraints, so it is an underestimate, but we use it as a proxy because it correlates with the true fuel cost when CA is included. To derive the fuel matrix F

for a given initial and final formation pair, we solve a Trajectory Optimization (TO) problem cast as a linear program. Each satellite i, in a decentralized manner, evaluates the fuel cost to move from its current location to each of the final locations in the formation by solving the Trajectory Optimization (TO) problem:

$$\textbf{TO problem to find } F: f_{ij} = \min_{u_i} \sum_{k=0}^{K-1} \|u_i(k)\|_1 \text{ subject to} \tag{6}$$

$$\textbf{boundary constraints: } x_i(0) = [\mathbf{x}_{\text{ini}}]_i,\ x_i(K) = [\mathbf{x}_{\text{des}}]_j, \tag{7}$$

$$\textbf{state space constraints: } x_i(k+1) = A(k)x_i(k) + Bu_i(k),\ k = 0, 1, \cdots, K-1, \tag{8}$$

$$\textbf{Actuator saturation:} \quad -u_{\max} \le u_i(k) \le u_{\max}, \tag{9}$$

$$\textbf{Slew-rate constraints:} \quad u_{r,\min} \le u_i(k{+}1) - u_i(k) \le u_{r,\max}. \tag{10}$$

where f_{ij} denotes the optimal fuel cost for satellite i to move from its initial location to position $j = 1, \ldots, N$ in the final formation, and $u_i(k)$ is the control input bounded in magnitude and rate by $[-u_{\max}, u_{\max}]$ and $[u_{r,\min}, u_{r,\max}]$, respectively.Here, the state-space dynamics is given by (5), discretized with a sampling time of T_c. These calculations are performed in parallel by all satellites, collectively forming the fuel matrix $F = [f_{ij}]_{N\times N}$. Step 2 then takes F as input.

2. *Trajectory Assignment with Collision Avoidance (TACA) (Step 2)*

Having solved the decentralized TO problem in Section III.B.1, we obtain the fuel cost f_{ij} and the corresponding fuel-optimal trajectories $x_i(k)$ and actuation sequences $u_i(k)$ for transfers in which satellite i is assigned to terminal location j. For any two satellites i and q assigned to terminal locations j and p, respectively, we define the minimum separation along their trajectories as

$$d_{ijqp} := \min_{k\in\{0,\ldots,K\}} \left\| r_i(k) - r_q(k) \right\|_2, \qquad q > i,\ j \neq p,$$

where $x_i(K) = [\mathbf{x}_{\text{des}}]_j$ and $x_q(K) = [\mathbf{x}_{\text{des}}]_p$. By using $d_{ijqp} > R_{\text{safe}}, \forall i, j$, TACA problem then finds a *collision-free assignment* $i \longrightarrow j$ and $q \longrightarrow p$; in other words, assignments with smallest fuel cost that simultaneously avoid collisions for the duration of the maneuver. The TACA problem can thus be formulated as:

$$\textbf{TACA problem: } \begin{cases} \min\limits_{\mathbf{B}} \sum\limits_{i=1}^{N} \sum\limits_{j=1}^{N} f_{ij} b_{ij} \\ \text{s.t. } \textbf{TA constraints: } \sum\limits_{i=1}^{N} b_{ij} = \sum\limits_{j=1}^{N} b_{ij} = 1, b_{ij} = \{0, 1\}, \\ \textbf{CA constraints:} \quad b_{ij} + b_{qp} \le 1,\ \text{if } d_{ijqp} < R_{\text{safe}}, \end{cases} \tag{11}$$

where CA constraints based on separating hyperplanes transform the prohibited assignment with suitable affine formulation. However, enforcing CA as hard constraints can render the TACA problem infeasible when no assignment satisfies (11). To always obtain a solution while still prioritizing safety, we soften the CA constraints by introducing slack variables and penalizing them in the objective. If a collision-free assignment exists, the optimizer drives all slacks to zero (yielding a safe solution); otherwise, the nonzero slacks identify which satellite pairs must violate R_{safe} and by how much. Let us now introduce two four-dimensional matrices $\mathbf{H}, \mathbf{C}^{\text{vio}}$ where $h_{ijqp} = \mathbf{H}[i,j,q,p], c^{\text{vio}}_{ijqp} = \mathbf{C}^{\text{vio}}[i,j,q,p]$ are entries at index (i,j,q,p).The modified TACA problem with softened CA constraints can thus be rewritten as

$$\textbf{Modified TACA problem:} \begin{cases} \min\limits_{\mathbf{B},\mathbf{H}} \sum\limits_{i=1}^{N} \sum\limits_{j=1}^{N} f_{ij} b_{ij} + c^{\text{vio}}_{ijqp}\, h_{ijqp}, \forall q,p \in \{1,2,\cdots,N\},\ q \neq i,\ p \neq j \\ \text{s.t. TA constraint in (11)}, \\ \textbf{CA constraints: } b_{ij} + b_{qp} \leq 1 + h_{ijqp},\ \text{if } c^{\text{vio}}_{ijqp} \geq 0, \\ h_{ijqp} \in \{0,1\}, \end{cases} \tag{12}$$

where $c^{\text{vio}}_{ijqp} = R_{\text{safe}} - d_{ijqp}$ and h_{ijqp} is a binary variable that selects an assignment based on minimal violation from R_{safe}.

3. Trajectory Optimization (TO) (Step 3)

With the resulting assignment $\mathbf{B} = [b_{ij}]_{N\times N}$ from solving the Modified TACA in (12) (Step 2), we finally solve the decentralized TO problem with a finer sampling time of T_f:

$$\begin{aligned} &\min_{u_i,x_i} \sum_{k=0}^{K-1} \|u_i(k)\|_1 \\ &\text{s.t. } x_i(0) = [\mathbf{x}_{\text{ini}}]_i, \quad x_i(K) = [\mathbf{x}_{\text{des}}\mathbf{B}]_i, \quad (8)-(10). \end{aligned} \tag{13}$$

to compute the optimal satellite trajectories $x_i^*(k)$ and actuation $u_i^*(k),\ \forall k$.

4. Shrinking-horizon Model-predictive Control (SHMPC) (Step 4)

The effects of unmodeled disturbances and modeling uncertainties need to be explicitly accounted for. In this work, we implement the TO algorithm in closed loop via an SHMPC formulation [66] to iteratively mitigate the effects of such uncertainties by evaluating a new control input over a prediction horizon based on the current state information. For a fixed final formation and time, SHMPC successively computes an optimal actuation command in a fully decentralized manner which can steer the actual satellite from its current location to its final location at t_f. With the update of new state information, the prediction horizon window shrinks as the start time of the TP algorithm gradually approaches to t_f. This method of successive prediction and reoptimizing the trajectory planner with more accurate initial positions thus yields a more robust control law that eventually renders a small trajectory prediction error in the face of modeling uncertainties and ignored perturbations.

```
Algorithm 2: Robust control sequence for trajectory planning
  Result: Robust control sequence $u^R(k)$ for $k \in \{0,\dots,K-1\}$
  Input: $\mathbf{x}_{\text{ini}}$, $\mathbf{x}_{\text{des}}$, $u_{\max}$, $u_{\min}$, $u_{r,\min}$, $u_{r,\max}$, $t_f$; sampling time $T_f$; $K$+1 grid points on $\{0,\dots,K\}$; number of
    windows $\beta > 0$; window length $K_1 = \frac{K+1}{\beta} \in \mathbb{N}$.
  Let $\mathbf{x}(k) \in \mathbb{R}^{6\times N}$ denote the formation state at step $k$. Initialize $c \leftarrow 1$, $\alpha \leftarrow 0$, $\mathbf{x}(0) \leftarrow \mathbf{x}_{\text{ini}}$, $\mathbf{x}(K) \leftarrow \mathbf{x}_{\text{des}}$.
  do
    $[\mathbf{x}^*, \mathbf{u}^*] \leftarrow$ solution of TO over $k \in [\alpha, K]$ using the latest available state estimate.
    Execute the closed loop over $k \in [\alpha,\ cK_1 - 1]$: apply $\mathbf{u}^*$ through the CL controller and let the system evolve
      in real flight between replanning steps, starting from the current state $\mathbf{x}(\alpha)$.
    $\mathbf{x}_{\text{ini}} \leftarrow \mathbf{x}(cK_1 - 1)$.
    For $k \in \{\alpha,\dots,cK_1 - 2\}$ set $u^R(k) \leftarrow u^*(k)$.
    $\alpha \leftarrow cK_1 - 1$;     $c \leftarrow c + 1$.
  while $c \le \beta$;
  return $u^R(k)$ for $k \in \{0,\dots,K-1\}$.
```

Assume the interval $[0, t_f]$ is discretized into $K+1$ samples, indexed by $k \in \{0,\dots,K\}$, with $\mathbf{x}(0) = \mathbf{x}_{\text{ini}}$ and $\mathbf{x}(K) = \mathbf{x}_{\text{des}}$. Choose an integer β denoting how many times the optimization is re-run during the maneuver; this yields the prediction interval $K_1 = \dfrac{K+1}{\beta}$. The more frequently the optimization is run (*i.e.*, the higher β), the smaller the trajectory prediction error. Algorithm 2 summarizes the procedure; for details, see [48].

In summary, this section developed an uncertainty-aware trajectory planner that makes the fundamentally large and nonconvex multi-satellite optimal planning problem tractable at scale. Key innovations that enable this are (*i*) recognizing that the linearized J_2 model exhibits sufficiently small prediction errors to permit a convex optimization, in which the state dynamics appear as affine equality constraints, without sacrificing accuracy, (*ii*) relaxing the CA constraints to reduce the communication burden while delegating guaranteed safe flight to the real-time controller, (*iii*) decoupling TA from TO, which avoids the complexity of a coupled formulation while sacrificing little fuel optimality, and (*iv*) introducing SHMPC to explicitly address modeling errors and disturbances.

IV. Control Layer

This section presents the control layer, which, as shown in Fig. 1, forms the inner layer of the architecture and operates at faster time scales. It consumes the optimal trajectories generated by the planning layer, together with local GPS and inter-satellite measurements. Within this layer, the estimator fuses these measurements into robust relative-state estimates that the real-time controller uses to compute safe, fuel-efficient thrust commands, while also providing the planning layer with accurate initial conditions for re-running the trajectory planner and, when necessary, triggering reconfiguration.

A. Real-time Controller

Building on the optimal trajectories from the previous section, this section details the real-time actuation on each satellite. The control logic must simultaneously achieve trajectory tracking, collision avoidance, and fuel-efficient operation. The control law derived herein relies on a state estimate $\hat{x}_i$, which we assume is provided by a dedicated estimator. The specifics of the estimation framework are presented subsequently in Section IV.B.

The controller works with three distinct sampling intervals: (*i*) TP providing $\{x_i^*, u_i^*\}$ every T_f, (*ii*) the estimator updating $\hat{\boldsymbol{x}}_i$ every T_e, and (*iii*) the controller executing at a rate of T_r. In this section, the controller operates with a sampling time of T_r. At each control step k, the planned trajectory $\{x_i^*(k), u_i^*(k)\}$ is obtained by linearly interpolating the slower planner outputs, and the state estimate $\hat{x}_i(k)$ is taken as the most recent output from the faster estimator. Finally, at each time step k, the satellite is assumed to have access to a set $O_i(k) \subset \mathbb{R}^3$ containing the minimum distances to all obstacles within its detection radius R_{det}, as provided by its sensors.

Given the setup and the assumptions outlined above, the real-time control problem is formally defined next.

Problem Statement 3.

Given: *Interpolated trajectory-planner outputs* $\{\mathbf{x}_i^*(k),\ \mathbf{u}_i^*(k)\}$ *with* $\mathbf{x}_i^*(k) = \mathrm{col}\big(\mathbf{r}_i^*(k),\ \mathbf{v}_i^*(k)\big)$*; the estimated state* $\hat{\mathbf{x}}_i(k) = \mathrm{col}\big(\hat{\mathbf{r}}_i(k),\ \hat{\mathbf{v}}_i(k)\big)$ *evaluated at the controller update times; and the obstacle set* $O_i(k)$*, for each satellite i.*

Find: *A real-time actuation command* $\mathbf{u}_i(k)$.

Such that:

(i) ***Tracking:*** $\mathbf{u}_i(k)$ *enables satellite i to track* $\mathbf{r}_i^*(k)$ *as closely as possible.*

(ii) ***Collision avoidance:*** *For all k, collision avoidance is enforced via* $\|\hat{\mathbf{r}}_i(k) - \mathbf{r}^o\|_2 \geq R_{\text{safe}}, \quad \forall\, \mathbf{r}^o \in O_i(k)$.

(iii) ***Decentralization:*** *The controller operates in a decentralized manner on each satellite, making it suitable for large swarms.*

Next, the high-level architecture of our proposed real-time controller is presented. The controller operates in two distinct modes: (*i*) an open-loop mode, which directly applies the TP command $u_i^*(k)$, and (*ii*) an artificial potential field (APF) mode, which generates a command $u_i^{\text{APF}}(k)$ to ensure real-time collision avoidance and safety when deviations from the TP trajectory arise due to disturbances, modeling mismatch, or unexpected obstacles. The operation modes are selected using a binary switch $c_i(k) \in \{0, 1\}$ and a latching mechanism. The control logic proceeds according to the following steps at each time instant k:

Step 1: Mode Selection: Determine the binary variable $c_i(k)$ based on tracking error and obstacle proximity. Specifically, define the tracking error and minimum obstacle distance as $\hat{e}_i(k) := \left\|\hat{r}_i(k) - r_i^*(k)\right\|$, and

$d_i^{\text{obs}}(k) := \min_{r^o \in O_i(k)} \left\|\hat{r}_i(k) - r^o\right\|$, then,

$$c_i(k) = \begin{cases} 1, & \hat{e}_i(k) \le R_{\text{APF}} \text{ and} \\ & d_i^{\text{obs}}(k) \ge R_{\text{safe}} \text{ and} \\ & \big(c_i(k-1) = 1 \text{ or we received a new trajectory from TP}\big), \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Step 2: Command Execution: Compute the actual control input $u_i(k)$ according to:

$$u_i(k) = \begin{cases} u_i^*(k), & \text{if } c_i(k) = 1, \\ u_i^{\text{APF}}(k), & \text{if } c_i(k) = 0. \end{cases} \tag{15}$$

Thus, the controller tracks the TP command only in open-loop mode. Once the APF-based safety command is invoked, it persists for the remainder of the planning cycle. When the controller switches from open-loop to APF control, then it cannot change until the higher-level TP produces a new trajectory. This prevents hazardous re-entry into the optimal tracker and oscillation between modes. The condition $c_i(k-1) = 1$ enforces a latch: once c_i becomes 0 at some step, it latches to 0 (the controller stays in safety mode) and remains there until a new trajectory explicitly sets $c_i(k)$ back to 1.

The remaining task is to compute $u_i^{\text{APF}}(k)$. The APF (attractive and repulsive) is first defined, followed by the real-time control law that generates $u_i^{\text{APF}}(k)$. Attractive potential fields Φ^{att} are often modeled using Gaussian, harmonic, or quadratic functions [31, 67]. Here, differentiability is required so that $\nabla\Phi_i^{\text{att}}$ is well defined, so a quadratic map $\Phi_i^{\text{att}} : \mathbb{R}^3 \longrightarrow \mathbb{R}$ is adopted:

$$\Phi_i^{\text{att}}(k) = \tfrac{1}{2}\big(\hat{r}_i(k) - r_i^*(k)\big)^\top K_{\text{att}} \left(\hat{r}_i(k) - r_i^*(k)\right). \tag{16}$$

where $K_{\text{att}} \in \mathbb{R}^{3\times 3}$ is a tuning matrix. Radial unboundedness is not a concern here, as the controller is assumed to keep a satellite sufficiently close to the goal. For the repulsion field, several differentiable choices exist (e.g., inverse-distance or exponential forms). It is required that the repulsive field vanishes whenever the satellite is outside the detection radius to avoid steady-state offsets. Thus, the inverse-distance form presented in [68] is selected. That is,

$$\Phi_i^{\text{rep}}(r^o, k) = \begin{cases} \frac{1}{2} K_{\text{rep}} \left(\dfrac{1}{\|\hat{r}_i(k) - r^o\|} - \dfrac{1}{R_{\text{det}}} \right)^2, & \text{if } \|\hat{r}_i(k) - r^o\| < R_{\text{det}}, \\ 0, & \text{otherwise,} \end{cases} \tag{17}$$

where $K_{\text{rep}} \in \mathbb{R}_{>0}$ is a tuning coefficient and $r^o \in O_i(k)$. The APF for satellite i is then the superposition of the attractive field and all active repulsive fields. This amounts to

$$\Phi_i(k) \;=\; \Phi_i^{\text{att}}(k) \;+ \sum_{r^o \in O_i(k)} \Phi_i^{\text{rep}}(r^o, k)\,. \tag{18}$$

Now, $u_i^{\text{APF}}(k)$ is determined from $\Phi_i(k)$. APF-based controllers commonly drive the system along the negative gradient of the potential. A classical approach treats the APF as a Lyapunov function and chooses the control to keep its time derivative negative, which moves the system toward the (moving) target and away from obstacles. However, this does not, in general, make the velocity equal to the negative gradient, so convergence to the target in the absence of obstacles may occur along a nonoptimal path. The work in [69], addresses this by enforcing the satellite's velocity to align with the negative gradient of the APF. This requires introducing a new Lyapunov function, $\mathcal{V}_i$, based on the velocity $v_i(k)$, which reaches its global minimum when $v_i(k) = -\nabla\Phi_i(k)$ and is defined by

$$\mathcal{V}_i(\boldsymbol{v}_i, k) \;=\; \tfrac{1}{2}\,\big\|\nabla\Phi_i(k) + \boldsymbol{v}_i\big\|_2^2. \tag{19}$$

Differentiating (19) gives $\dot{\mathcal{V}}_i = \big(\nabla\Phi_i(k) + v_i\big)^\top \Big(\mathcal{H}\big(\Phi_i(k)\big)\, v_i + \dot{v}_i\Big)$. Here $\dot{v}_i$ denotes the acceleration of the satellite i and is selected to ensure that $\dot{\mathcal{V}}_i$ is always negative, as follows:

$$\dot{v}_i \;=\; -\mathcal{H}\big(\Phi_i(k)\big)\, v_i \;-\; K_d\big(\nabla\Phi_i(k) + v_i\big),$$

where selecting $K_d \in \mathbb{R}^{3\times3}$ as a positive definite matrix ensures $\dot{\mathcal{V}}_i \le 0$, driving $\boldsymbol{v}_i \to -\nabla\Phi_i$. Now that we know the acceleration of satellite i, we can translate this into the actuation command $u_i^{\text{APF}}(k)$; that is, find the actuation command that will produce this acceleration by plugging in the value of $\dot{v}_i$, substituting estimated-state values of $\hat{r}_i$ and $\hat{v}_i$ for r_i and v_i, then solving the nonlinear dynamics equations for u_i. Finally, set $u_i^{\text{APF}}(k) = u_i$. More specifically,

$$u_i^{\text{APF}}(k) = -\,\mathcal{H}\big(\Phi_i(k)\big)\,\hat{\boldsymbol{v}}_i(k) - K_d\big(\nabla\Phi_i(k) + \hat{\boldsymbol{v}}_i(k)\big) - \mathcal{G}\big(\hat{r}_i(k),\, \hat{\boldsymbol{v}}_i(k),\, kT_r\big), \tag{20}$$

where $\mathcal{G}$ is given in (2).

To balance tracking accuracy and fuel usage when the APF mode is active, the APF gains $K_{\text{att}}, K_{\text{rep}}, K_d$ can be tuned for minimal position-tracking error, control effort, and proximity to obstacles; the reader is referred to [49] for details of the tuning optimization.

In summary, this section developed a real-time controller that tracks the planned trajectories while maintaining collision avoidance. Key innovation that enable this is the introduction of a switching control architecture that applies optimal commands from the TP in an open-loop manner when the formation is safely separated and on-track and activates the APF mode only near safety limits, so that we retain APF's fast, low-computation safety response while avoiding its long-term lack of fuel optimality.

B. Robust Estimation

The state estimator must run on each satellite, producing robust local state estimates for both the CL and the TP from PL, while incorporating information received from GPS and neighboring satellites when available. To establish the state estimator consider the following assumptions: (*i*) Each satellite S_i carries a GPS receiver that returns position measurements y_i^{GPS} with a known, satellite-specific noise covariance R_i^{GPS}. While the nominal noise is modeled as zero-mean with covariance R_i^{GPS}, the measurements may occasionally contain unmodeled outliers due to atmospheric effects or transient hardware errors, (*ii*) Each satellite is also equipped with an inter-satellite sensor that provides relative-position measurements for neighbors within $\min\{R_{\mathrm{det}}, R_{\mathrm{comm}}\}$ with nominal, fixed covariance R_i^{rel}. (*iii*) The formation is modeled as a connected, time-varying graph where satellites are nodes, and an undirected edge exists between two satellites if and only if their separation is at most $\min\{R_{\mathrm{det}}, R_{\mathrm{comm}}\}$. (*iv*) At estimator time-step k for satellite S_i, although every satellite is equipped with GPS, inter-satellite ranging is also used to construct multiple path-based relative position measurements. . Let $\mathcal{T}(\mathrm{S}_i, k)$ denote the set of all simple paths (i.e., those with no repeated vertices) starting at S_i. For any $\ell \in \mathcal{T}(\mathrm{S}_i, k)$, let $\mathcal{T}_\ell'(\mathrm{S}_i, k)$ be the ordered list of nodes on ℓ after S_i, and $f_\ell(\mathrm{S}_i, k)$ its terminal node. Using the GPS at $f_\ell(\mathrm{S}_i, k)$ together with the inter-satellite ranges along $\mathcal{T}_\ell'(\mathrm{S}_i, k)$, a relative position measurement of S_i is constructed for path ℓ and made available to S_i, denoted $Y_{i,\ell}^N(k) \in \mathbb{R}^3$. The problem statement of this section is given next.

Problem Statement 4.

Given: *GPS position measurements* $\mathbf{y}_i^{\mathrm{GPS}}(k) \in \mathbb{R}^3$*; path-based position measurements* $\{y_{i,\ell}^{rel}(k)\}_{\ell\in\mathcal{T}(\mathrm{S}_i,k)} \subset \mathbb{R}^3$ *constructed from inter-satellite ranges along* $\mathcal{T}_\ell'(\mathrm{S}_i, k)$ *and the GPS at* $f_\ell(\mathrm{S}_i, k)$*; and the sampled control* $\hat{\mathbf{u}}_i(k) \in \mathbb{R}^3$ *(zero-order hold of* $\mathbf{u}_i(k)$ *with sampling step size* T_e*).*

Find: *The state estimate* $\hat{\mathbf{x}}_i(k)$ *and its error covariance* $P_i(k)$ *at each estimator sample (every* T_e*).*

Such that:

(i) ***Nominal accuracy:*** *The estimator provides accurate state estimates under nominal measurement conditions.*

(ii) ***Outlier robustness:*** *In the presence of bounded measurement outliers in GPS and relative measurements, the estimation error and covariance remain bounded.*

(iii) ***Communication efficiency:*** *The estimator requires minimal inter-satellite communication.*

The proposed estimator addresses this problem in two steps:

Step 1: Measurement acquisition: GPS and inter-satellite measurement models are provided such that they explicitly capture the statistical dependencies and covariance structure of these signals.

Step 2: Robust state estimation: Using the time-varying model in (5), a robust state estimator is proposed to fuse the available measurements while mitigating outliers.

A Kalman filter structure is chosen for the state estimator to address the problem statement. The prediction model used in the estimator is the linearized J_2 model introduced in (5) with a sampling time of T_e. GPS measurement noise is modeled as a single zero-mean Gaussian. This is a reasonable assumption because the nominal error sources, receiver thermal noise, residual clock errors, broadcast ephemeris/clock uncertainties, and imperfect neutral-/ionospheric corrections are well approximated by Gaussian distributions. Therefore, the GPS measurement of satellite S_i at time-step k is

$$y_i^{\mathrm{GPS}}(k) = r_i(k) + n_i^{\mathrm{GPS}}(k), \text{ with } n_i^{\mathrm{GPS}}(k) \sim \mathcal{N}\big(\mathbf{0},\, R_i^{\mathrm{GPS}}\big). \tag{21}$$

For relative sensing, let $\ell \in \mathcal{T}(\mathrm{S}_i, k)$ be a simple path starting at S_i, with ordered nodes $\mathcal{T}_\ell'(\mathrm{S}_i, k)$ after S_i and terminal node $f_\ell(\mathrm{S}_i, k)$. Prior formation-flying studies model inter-satellite relative measurement noise as a single Gaussian [70]. Thus, the relative measurement of $r_i(k)$ induced by path ℓ is

$$y_{i,\ell}^{\mathrm{rel}}(k) = r_i(k) + n_{f_\ell(\mathrm{S}_i,k)}^{\mathrm{GPS}}(k) + \sum_{S_j \in \mathcal{T}_\ell'(\mathrm{S}_i,k)} n_j^{\mathrm{rel}}(k), \quad n_j^{\mathrm{rel}}(k) \sim \mathcal{N}\big(\mathbf{0},\, R_j^{\mathrm{rel}}\big), \tag{22}$$

with n_j^{rel} and $n_{f_\ell}^{\mathrm{GPS}}$ mutually independent noise processes across nodes and paths. The covariance of the relative measurement is described as

$$\mathrm{Cov}\big(y_{i,\ell}^{\mathrm{rel}}(k)\big) = R_{f_\ell(\mathrm{S}_i,k)}^{\mathrm{GPS}} + \sum_{S_j \in \mathcal{T}_\ell'(\mathrm{S}_i,k)} R_j^{\mathrm{rel}}. \tag{23}$$

The set of all relative measurements obtained by satellite S_i through inter-satellite communication at time-step k is then defined as

$$\mathcal{Y}_i^{\mathrm{rel}}(k) = \Big\{ y_{i,\ell}^{\mathrm{rel}}(k) \;\Big|\; y_{i,\ell}^{\mathrm{rel}}(k) = r_i(k) + n_{f_\ell(\mathrm{S}_i,k)}^{\mathrm{GPS}}(k) + \sum_{S_j \in \mathcal{T}_\ell'(\mathrm{S}_i,k)} n_j^{\mathrm{rel}}(k) \Big\}, \quad \forall \ell \in \{1, \ldots, |\mathcal{T}(\mathrm{S}_i, k)|\}. \tag{24}$$

Stacking the elements of $\mathcal{Y}_i^{\mathrm{rel}}(k)$, the vector of relative measurements is $y_i^{\mathrm{rel}}(k) = \mathrm{col}\big(y_{i,1}^{\mathrm{rel}}(k),\, \ldots,\, y_{i,|\mathcal{T}(\mathrm{S}_i,k)|}^{\mathrm{rel}}(k)\big)$. In then follows that, combining GPS and inter-satellite measurements, the redundant measurement vector for S_i is

$$y_i(k) = \mathrm{col}\big(y_i^{\mathrm{GPS}}(k),\, y_i^{\mathrm{rel}}(k)\big). \tag{25}$$

Naturally, the relative measurement vector $y_i(k)$ may contain correlated components since paths in $\mathcal{T}(\mathrm{S}_i, k)$ can overlap and share sensors. Considering two distinct relative measurements $y_{i,\ell_1}^{\mathrm{rel}}(k)$ and $y_{i,\ell_2}^{\mathrm{rel}}(k) \in \mathcal{Y}_i^{\mathrm{rel}}(k)$, their cross-covariance is

$$\mathrm{Cov}\big(y^{\mathrm{rel}}_{i,\ell_1}(k),\, y^{\mathrm{rel}}_{i,\ell_2}(k)\big) = \underbrace{\sum_{S_j \in \left(\mathcal{T}'_{\ell_1}(\mathrm{S}_i,k)\cap\mathcal{T}'_{\ell_2}(\mathrm{S}_i,k)\right)} R^{\mathrm{rel}}_j}_{\text{shared-path relative noise}} + \begin{cases} R^{\mathrm{GPS}}_{f_{\ell_1}(\mathrm{S}_i,k)}, & f_{\ell_1}(\mathrm{S}_i,k) = f_{\ell_2}(\mathrm{S}_i,k), \\ 0, & f_{\ell_1}(\mathrm{S}_i,k) \neq f_{\ell_2}(\mathrm{S}_i,k). \end{cases} \tag{26}$$

From (25) and (26), the covariance of the redundant measurement vector $y_i(k)$, dropping the (k) for conciseness, is then

$$\mathrm{Cov}(y_i(k)) = \begin{bmatrix} R^{\mathrm{GPS}}_i & 0 & 0 & \cdots & 0 \\ 0 & R^{\mathrm{rel}}_{i,1} & \mathrm{Cov}\big(y^{\mathrm{rel}}_{i,1}, y^{\mathrm{rel}}_{i,2}\big) & \cdots & \mathrm{Cov}\big(y^{\mathrm{rel}}_{i,1}, y^{\mathrm{rel}}_{i,|\mathcal{T}(\mathrm{S}_i,k)|}\big) \\ 0 & \mathrm{Cov}\big(y^{\mathrm{rel}}_{i,1}, y^{\mathrm{rel}}_{i,2}\big) & R^{\mathrm{rel}}_{i,2} & \cdots & \mathrm{Cov}\big(y^{\mathrm{rel}}_{i,2}, y^{\mathrm{rel}}_{i,|\mathcal{T}(\mathrm{S}_i,k)|}\big) \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \mathrm{Cov}\big(y^{\mathrm{rel}}_{i,1}, y^{\mathrm{rel}}_{i,|\mathcal{T}(\mathrm{S}_i,k)|}\big) & \mathrm{Cov}\big(y^{\mathrm{rel}}_{i,2}, y^{\mathrm{rel}}_{i,|\mathcal{T}(\mathrm{S}_i,k)|}\big) & \cdots & R^{\mathrm{rel}}_{i,|\mathcal{T}(\mathrm{S}_i,k)|} \end{bmatrix}. \tag{27}$$

Thus, the output equations for the i-th satellite is

$$y_i(k) = C_i(k)\, x_i(k) + n_i(k), \tag{28}$$

where $n_i(k)$ collects all measurement noises available to satellite S_i at time k and $C_i(k)$ is the augmented measurement matrix and depends on the total number of measurements available to satellite S_i. That is,

$$C_i(k) = I_{|\mathcal{T}(\mathrm{S}_i,k)|+1} \otimes \begin{bmatrix} I_3 & 0_{3\times 3} \end{bmatrix}. \tag{29}$$

In large formations, the number of path-based measurements and the size of the matrix C_i can grow combinatorially. Also, since longer paths tend to be noisier, it is beneficial to limit the length of paths used. To avoid collecting unnecessary measurements and incurring excessive communication overhead, a user-defined outlier tolerance n^* is introduced. Hence, a measurement-acquisition algorithm is generated that gathers enough data to tolerate up to n^* outliers while keeping communication overhead minimal. Let S_j be an n-th order neighbor of S_i if a path of length n connects them. The measurement acquisition procedure in Algorithm 3 increases the neighborhood order until the user-defined outlier tolerance n^* is satisfied, minimizing communication while preserving estimator performance.

With the full measurement set at time k assembled, the next step is estimation under the time-varying model in (5). Two paths are standard. In (*i*), we adopt a Robust Generalized Maximum-Likelihood Kalman Filter (RGMKF) following prior work (e.g., [50]), wherein the available measurements are first augmented with the state prediction, pre-whitened to remove correlations and normalize units, and used to form robust residuals; a GM objective is then

Algorithm 3: Measurement acquisition for satellite S_i (Step 1 of the estimation problem in Section IV.B), minimizing inter-satellite communication.

```
Input: $n^*, i, k$
Output: $y_i(k)$
Initialize $n = 1$
while $\lfloor \frac{|\mathcal{T}(S_i,k)|-1}{2} \rfloor \leq n^*$ do
    Collect measurements from $n$th-order neighbors
    Update $y_i(k)$ with new measurements
    $n = n + 1$
end
```

minimized via Iteratively Reweighted Least Squares (IRLS) until convergence, yielding a corrected state estimate and covariance consistent with the dynamics in (5). In (*ii*), the same pre-whitening step is used solely to identify and discard outlying measurements, after which a conventional Kalman filter is applied to the cleaned measurement set. The choice between (*i*) and (*ii*) depends on expected outlier prevalence and computational budget: heavier contamination favors the fully robust RGMKF, while largely reliable data and tighter resources favor outlier rejection followed by standard filtering. In summary, this section developed an estimator that provides reliable relative-state estimates under noisy and possibly outlying measurements. Key innovations that enable this are (*i*) leveraging redundancy in satellites and onboard sensors to fuse inter-satellite and GPS measurements, with the formation planner indirectly supporting the estimator by improving network reliability, which improves measurement availability and reliability for each satellite, and (*ii*) using an outlier-tolerant Kalman-filter update to handle corrupted GPS measurements.

V. High-Fidelity Simulation and Hardware-in-the-Loop Validation

In this section, we first evaluate the proposed GNC framework in high-fidelity simulations of a CanX-4/5–inspired mission, and second, perform HIL testing of the TP (from PL) to verify implementability on flight hardware and to quantify its runtime. Note, we prioritized HIL testing of the TP (over FP) because its on-board runtime determines whether the swarm can regularly use fuel-optimal planned trajectories, if TP updates are too slow, the system relies more on the fallback APF commands, hurting mission efficiency. The FP runs only when a reconfiguration is triggered and produces a terminal geometry, so it can tolerate longer compute times, making its exact on-hardware runtime less critical to quantify.

A. High-Fidelity Simulation Mission

The mission considered in this section is inspired by the CanX-4/5 mission (University of Toronto Institute for Aerospace Studies, Space Flight Laboratory, 2014) [71]. The objective of the CanX-4/5 mission was to demonstrate precision autonomous formation flying with two nanosatellites. Although our GNC framework targets scalable multi-satellite formations rather than precision two-satellite flight, we benchmark against this two-satellite mission because its documentation is exceptionally comprehensive. Note, however, that differences in application and scale required several assumptions not specified in the CanX-4/5 documentation. In the following, we present the mission details in three parts: (*i*) environment assumptions, (*ii*) satellite assumptions, and (*iii*) mission requirements. For each point, we explicitly identify all deviations from and additions to the published CanX-4/5 setup.

Environment Assumptions: We assume a formation of 10 satellites in low Earth orbit at an altitude of approximately 660 km with orbital elements $i = 98.4437°$, $\Omega = 135.1155°$, $e = 0.0023$, and $\theta = 0°$ and orbital period of approximately 90 min. Formation flight is simulated using high-fidelity nonlinear dynamics as described in (2). To evaluate collision-avoidance capabilities of our GNC, we introduce environmental obstacles modeled as spherical keep-out zones, consistent with prior studies [72].

Compared with CanX-4/5, our study (*i*) considers a 10-satellite formation (vs. two), (*ii*) initializes the orbit from a CelesTrak TLE for CanX-4 at epoch 25 September 2025, (*iii*) employs high-fidelity nonlinear simulation rather than flight data, and (*iv*) explicitly includes obstacle-avoidance scenarios, which were not part of the CanX-4/5 demonstrations.

Satellite Assumptions: The CanX-4/5 nanosatellites use $20 \times 20 \times 20$ cm Generic Nanosatellite Bus with a total mass of approximately 15 kg. In our study, we adopt the same volume and mass. We further assume each satellite is equipped with the following sensors and subsystems:

- **Thrusters:** In the CanX-4/5 mission, each vehicle employed a liquefied cold-gas propulsion system with four independently controlled thrusters arranged in a cruciform on a single face, delivering 10 mN of thrust per thruster. For modeling in our study, we assume ideal bidirectional translational authority along each body axis: three virtual actuators apply perfectly collinear forces in each direction, abstracting a low-level thrust-allocation mapping from body-axis force commands to the thrusters. The maximum thrust in each direction is taken as 10 mN, consistent with the capability of the actual thrusters. We further assume that the directional thrusts produce no net torque; although not physically realizable due to off-axis components and alignment errors, the resulting moments are assumed to be rejected by an attitude controller, which is not treated in this work.
- **GPS:** Each CanX-4/5 spacecraft carried a NovAtel OEMV-1G receiver; similarly, we assume each satellite is equipped with GPS. While such receivers output position in an Earth-fixed frame that would be transformed to LVLH, we adopt the simplifying assumption that GPS provides position directly in LVLH. To evaluate estimator performance in simulation, measurement noise must be modeled explicitly (whereas in flight it is inherently present). In our estimator, we assume that measurement noise is drawn from a nominal Gaussian distribution (see (21)), which forms the basis for state estimation. To assess robustness against outliers, we extend the measurement-noise model, consistent with prior work showing GPS position errors are well modeled by Gaussian

mixtures [73–75], as

$$b \sim \text{Bernoulli}(p_{\text{out}}), \qquad n_i^{\text{GPS}}(k) \ \sim \ (1-b)\,\mathcal{N}(0, \sigma_{\text{nom}}^2 I_3) \ + \ b\,\mathcal{N}(0, \sigma_{\text{out}}^2 I_3),$$

where $p_{\text{out}} \in [0,1]$ is the probability of an outlier. The $\mathcal{N}(0, \sigma_{\text{nom}}^2 I_3)$ term represents the *nominal* noise with smaller variance, $\sigma_{\text{nom}} < \sigma_{\text{out}}$, while the $\mathcal{N}(0, \sigma_{\text{out}}^2 I_3)$ term represents the *outlying* noise associated with larger, infrequent deviations. Based on the OEMV-1G hardware datasheet [76], we set $\sigma_{\text{nom}} \approx 1.0$ m per axis, $\sigma_{\text{out}} = 5.0$ m, and $p_{\text{out}} = 0.10$ (in line with empirical GMM fits [73]).

- **Obstacle-identification sensor:** CanX-4/5 did not carry a proximity sensor. We assume a proximity sensor with a detection radius of $R_{\text{det}} = 100$ m to demonstrate autonomous obstacle avoidance, consistent with flight hardware on Orbital Express, TriDAR, and RemoveDEBRIS, which achieved comparable or greater ranges [77–79].
- **Inter-satellite distance measurement:** CanX-4/5 obtained inter-satellite distance measurements via carrier-phase differential GNSS baselines. This couples GPS errors into the inter-satellite distance measurement; however, we make the simplifying assumption that this coupling is absent. Specifically, we assume each satellite to carry an independent inter-satellite sensor providing 3D line-of-sight measurements to its neighbors, uncorrelated with GPS errors. In other missions, independent inter-satellite links (e.g., RF, laser, optical, or vision-based) explicitly decouple ranging from GPS (e.g., FFRF and TanDEM-X [70, 80]), so this is not an overly simplifying assumption. The measurement noise is modeled as in (22); we set the standard deviation R^{rel} to 0.04 m I_3, consistent with PRISMA's FFRF modeling [80].
- **Inter-satellite communications:** CanX-4/5 carried an ISL radio with two S-band patch antennas for bidirectional messaging. Although qualified for separations of several kilometers [37], in our simulations we assume a conservative effective range of $\sim$400 m and a binary link model (connected within range, disconnected otherwise) as a simplifying abstraction.

Mission Requirements: We design the simulation to be comparable in scale and objectives to the CanX-4/5 demonstration of a 500 m ATO reconfiguration. Unlike CanX-4/5, which involved two satellites, our study considers a formation of $N = 10$. Because formation geometry and initial conditions for multi-satellite cases are not available from the CanX-4/5 reports, we define these explicitly while preserving the same reconfiguration scale, namely a 1000 m $\rightarrow$ 500 m ATO transition. For the reference geometry $\mathbf{F}_0$, define $\alpha_i = \frac{2\pi(i-1)}{N}$. The i-th mission prescribed position is $[\mathbf{F}_0]_i = \text{col}\big(\frac{500}{2}\sin\alpha_i,\ 500\cos\alpha_i,\ 0\big)$, corresponding to a 500 m ATO. The initial positions at $t = 0$ use the same parameterization: $r_i(0) = \text{col}\big(\frac{1000}{2}\sin\alpha_i,\ 1000\cos\alpha_i,\ 0\big)$, representing a 1000 m ATO. Thus, the simulation mirrors the CanX-4/5 maneuver in magnitude; to also match the timescale , we set t_f to almost half an orbital period. Finally, we impose the following performance requirements:

- **Fuel usage:** total reconfiguration cost per satellite satisfies $\Delta v \leq 20$ m/s over the whole maneuver .
- **Estimation accuracy:** RMS relative-navigation error satisfies ≤ 1 m.
- **Tracking accuracy:** steady-state RMS relative error satisfies ≤ 10 m.
- **Obstacle avoidance:** trajectories remain collision-free throughout the maneuver.

Results: Here we present the results of the formation-flying mission simulation using our proposed GNC framework, from formation design by the formation planner through trajectory planning, execution, and real-time control and estimation under unexpected obstacles and measurement noise. The performance is evaluated against the key mission requirements described above.

Starting with the formation planner, we pass the formation $\mathbf{F_0}$ with 8 satellites ($M = 8$). Using a Monte Carlo estimator with 10^4 trials and edge operation probability $p = 0.9$, the initial formation's reliability is $\approx 59.1\%$ (left panel of Fig. 5). In the remaining subfigures, satellites 9 and 10 are added and optimally relocated using the algorithms of Section III.A, increasing the all-terminal reliability to 94.7% and then 98.6%. The desired formation $\mathbf{F}_{\text{des}}$ is updated accordingly; velocities are assigned as in Section III.A.2, and the resulting terminal state $\mathbf{x}_{\text{des}}$ is provided to the trajectory planner.

To assess whether jointly optimizing the placements of satellites 9 and 10 could yield a better terminal reliability than the proposed iterative addition, we also performed an exhaustive randomized search over the whole region. Specifically, in MATLAB we repeatedly sampled candidate pairs of relative positions for satellites 9 and 10 uniformly at random within the admissible space, formed the corresponding terminal graph, and evaluated all-terminal reliability via the same Monte Carlo simulations. Across a large number of samples, this batched random search did not find any placement pair that exceeded the reliability achieved by the iterative planner. Moreover, reaching a solution of comparable quality required substantially more samples and computation (on the order of $\sim 10\times$ the runtime of the iterative approach). While more sophisticated global-search heuristics could be explored in future work, these results suggest that the iterative planner attains near-best reliability in this scenario at a fraction of the computational cost.

The desired initial and terminal states from the FP are then provided to the trajectory planner, configured with $T_c = 250$ s and $T_f = 20$ s. The resulting fuel-optimal trajectories are shown in Fig. 6. All satellite trajectories reconfigure the maneuver from the initial to the desired formation while satisfying the actuator constraints.

Now, given the optimal trajectories from the TP, the real-time mission is executed as follows: we employ the real-time estimator in Section IV.B and the controller in Section IV.A, both running at 1 Hz. To demonstrate the framework's collision-avoidance capability, we place a spherical obstacle directly on the preplanned trajectory of one satellite as shown in Fig. 7. The figure also shows the actual paths taken by all satellites compared with the planned ones. During the simulation, the SHMPC is triggered three times to replan the optimal trajectory and maintain collision-free motion. All satellites apply the optimal actuation computed by the trajectory planner and exhibit good tracking. For the satellite

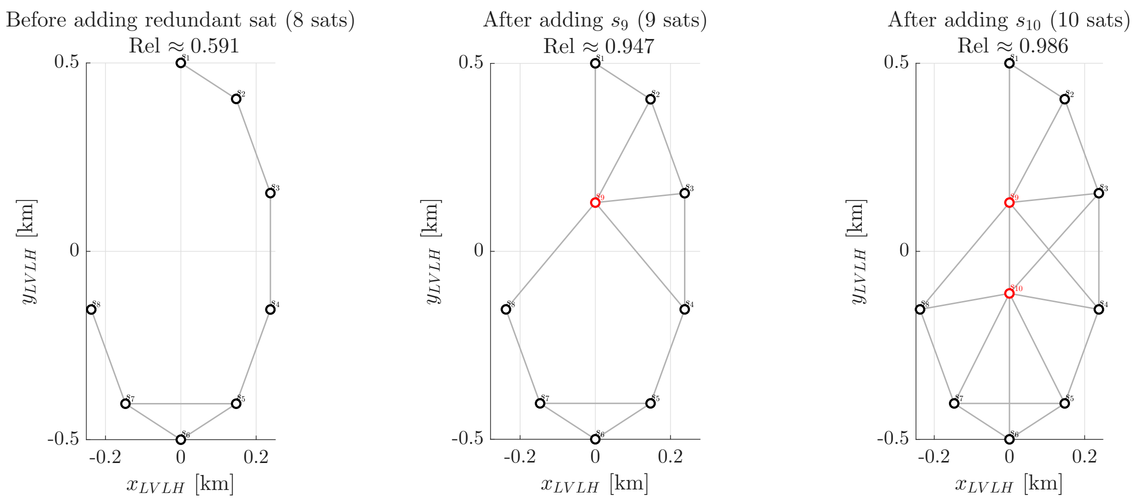


Fig. 5 Formation reconfiguration when adding two redundant nodes to a formation with eight prescribed positions.

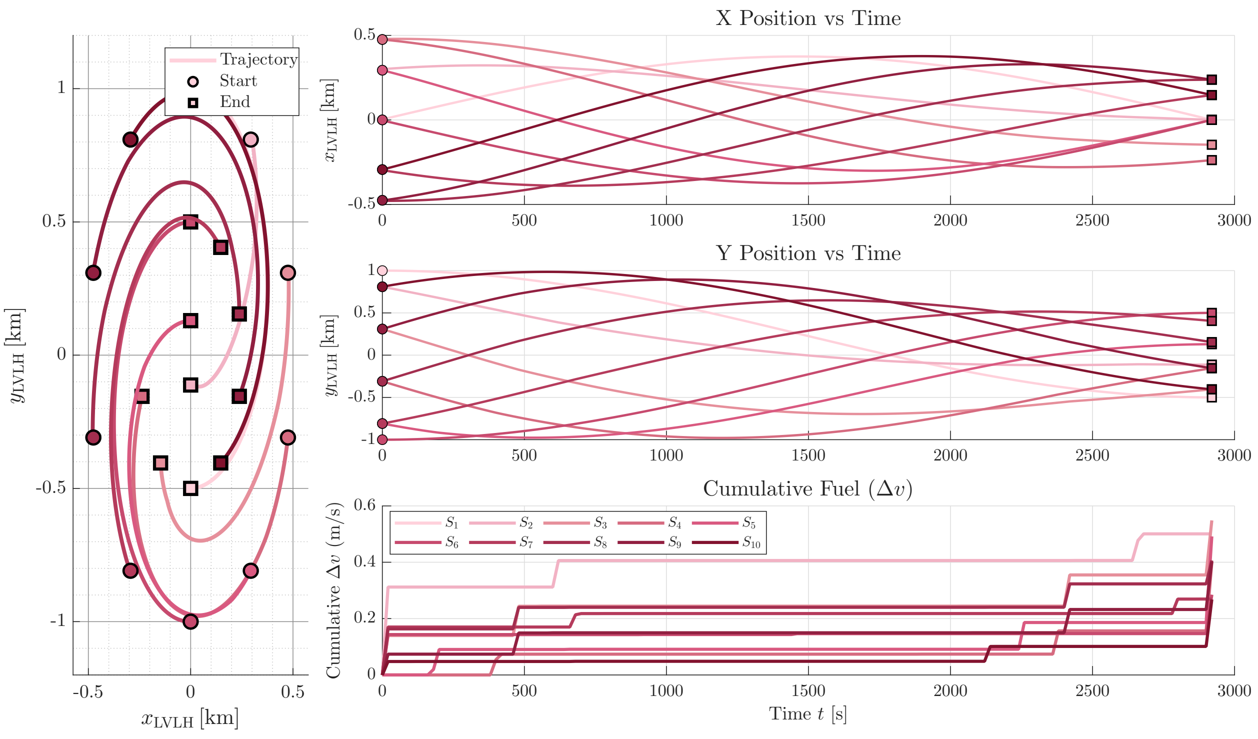


Fig. 6 TP planned trajectory. Left: projection of trajectories on the orbit plane; z_{LVLH} omitted due to small variation. Right: $x(t)$ and $y(t)$ (top, middle) and cumulative Δv (bottom).

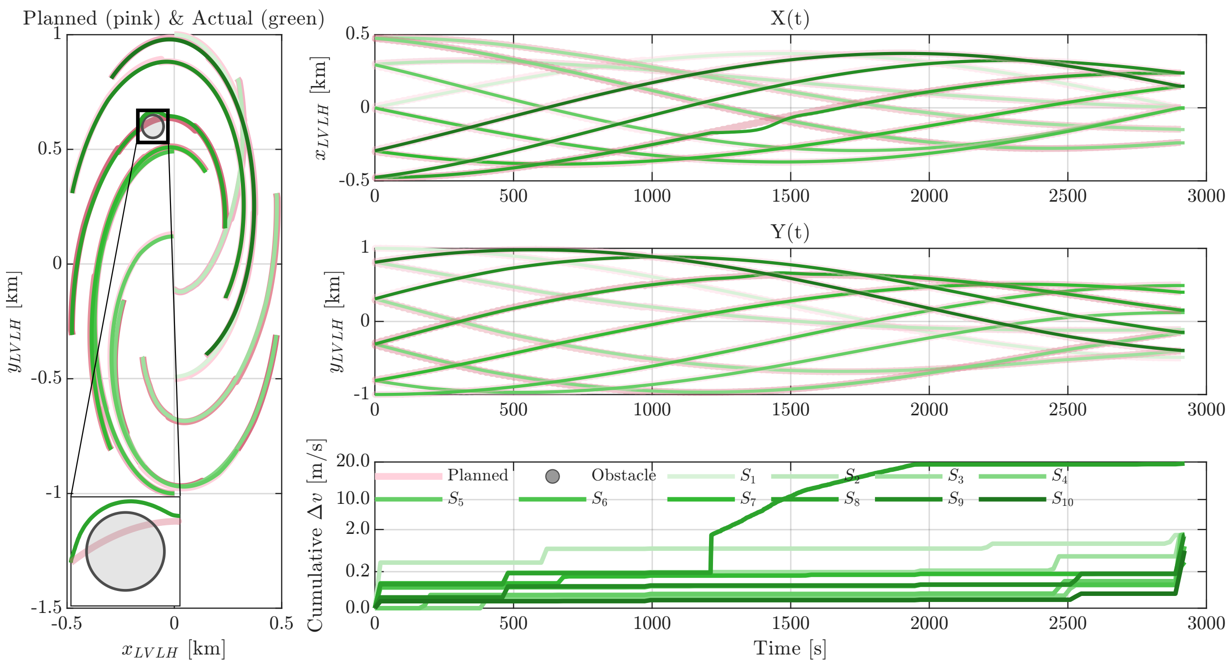


Fig. 7 Planned (pink, wide) vs. actual closed-loop (green) LVLH projected trajectories. A single LVLH-fixed keep-out zone (gray) intersects the references of one of satellites; upon detection, the APF safety mode detours them around the obstacle before rejoining the plan. Other vehicles track their references closely. The maneuver is collision-free and maintains the 50 m separation margin.

whose path intersects the obstacle, an event not accounted for by the TP, the APF controller is triggered, guiding the satellite around the obstacle while maintaining a safe distance. This validates the system's ability to react to unforeseen obstacles without centralized replanning. Finally, a performance summary is presented in Fig. 8, evaluated against the core mission requirements:

- **Tracking Accuracy:** The control tracking RMSE for the satellite encountering the unforeseen obstacle is significantly higher than for the others, as shown in the bar chart, due to the additional maneuvering required to avoid the obstacle. This higher RMSE is accompanied by greater fuel usage, which is expected given the deviation from the optimal path. For all other satellites, the tracking RMSE remains below approximately 10 m.
- **Fuel Usage (Δv):** Similar to tracking accuracy, the actual Δv for the satellite encountering the unforeseen obstacle increased significantly due to the avoidance maneuver. For the other satellites, the actual Δv closely matched the fuel-efficient planned values.
- **Collision Avoidance:** The minimum inter-satellite distance for all satellites remained above the 50 m safety threshold throughout the simulation, confirming that the system is collision-free.
- **Estimation Accuracy:** The estimation RMSE for all satellites remains approximately below 0.7 m throughout the

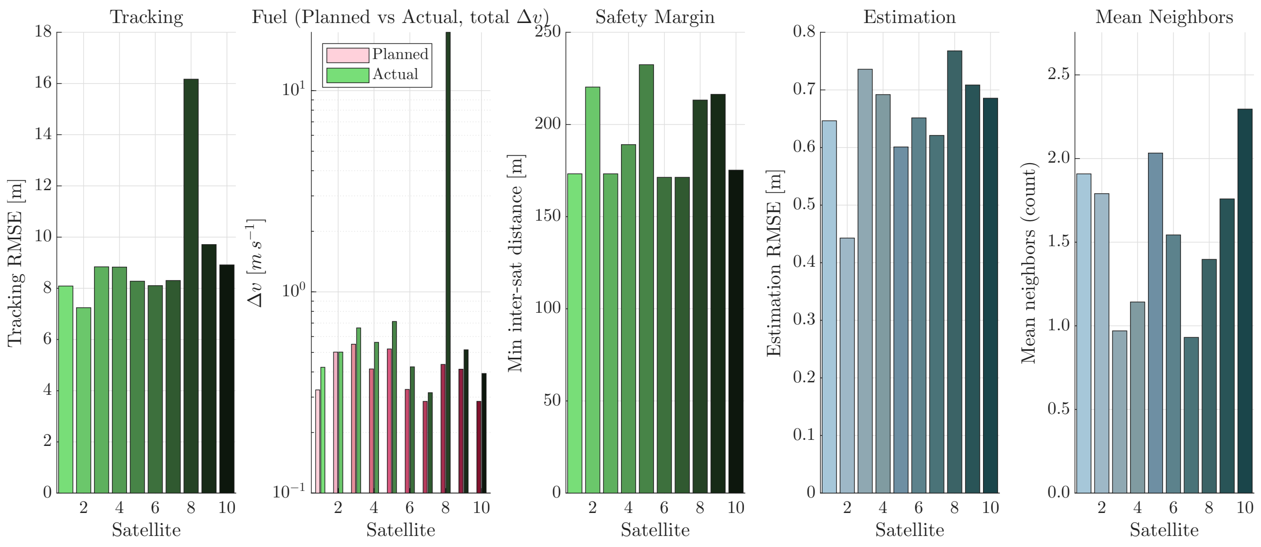


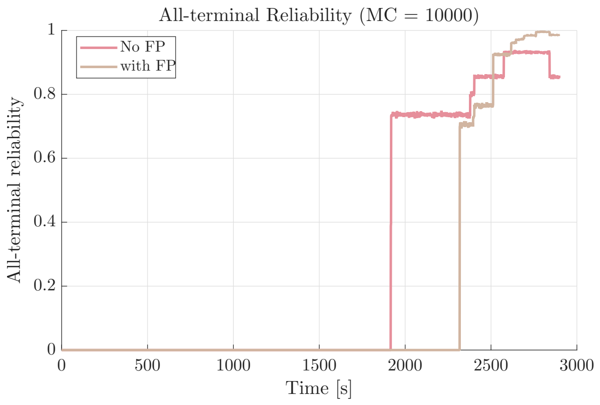
Fig. 8 Per-satellite closed-loop metrics. Panels (left→right): tracking RMSE; total Δv (actual vs. planned); minimum pairwise distance; estimation RMSE; mean neighbor count. S_8 show larger RMSE and higher Δv from obstacle-avoidance detours; others track closely and match the plan. All vehicles maintain $\geq 50\,\mathrm{m}$ separation. Estimation error accuracy trends with neighbor count (right).

simulation. The estimator's performance appears correlated with the mean number of neighboring satellites, as shown in the rightmost plot of Fig. 8, since a larger number of neighbors provides additional measurements to the estimator.

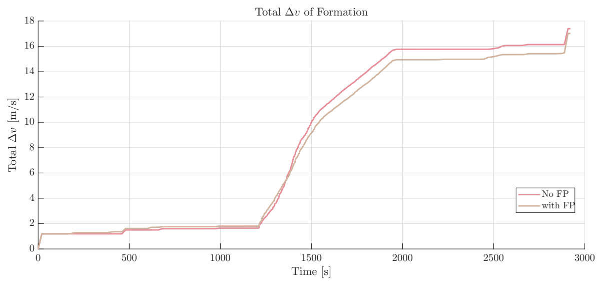
Finally, we circle back to our GNC design requirements: achieving both reliability and efficiency. Recall that we adopted a deliberate decoupling, reliability via the FP and efficiency via the TP, which sacrifices global optimality for tractability. To assess the effect of incorporating the FP itself, Fig. 9 plots total fuel consumption and all-terminal reliability over time for two cases: (*i*) the FP updates $\mathbf{x}_{\text{des}}$ before TP and real-time execution, and (*ii*) the reference formation is passed directly to the TP, bypassing the FP. With FP, the all-terminal reliability constraint is satisfied at the end of the maneuver but not enforced during the transient; without FP, reliability remains lower, and fuel consumption is not reduced. Even though no general conclusion can be drawn, these results suggest that incorporating the FP consistently improves reliability without compromising fuel efficiency, at least for this mission. Finally, while a joint reliability–efficiency optimization could yield a formation with comparable reliability at lower fuel than case (*i*), our decoupled framework meets the specified requirements.

B. Hardware-in-the-Loop (HIL) Validation of TP

This section reports HIL test results for the TP from the PL. The TP, as presented in Sec. III.B, comprises: (*i*) fuel-matrix calculation (Step 1), where each satellite solves a convex program to populate $F = [f_{ij}]$; (*ii*) TACA (Step 2), which solves a centralized mixed-integer program; and (*iii*) TO (Step 3), where each satellite, in a decentralized manner, solves for the optimal trajectory given its trajectory assignment. Note that Step 1 and Step 3 essentially solve the



(a) All-terminal reliability as a function of time for the NO-FP and FP runs.



(b) Cumulative total Δv of the entire formation over the mission time.

Fig. 9 Comparison of formation performance with and without the Formation Planner (FP): (a) all-terminal reliability over time and (b) cumulative fuel usage (total Δv) for the full formation.

same problem (i.e., the TO problem in (6)) but differ in two ways: (*i*) sampling rate: Step 1 discretizes the dynamics with a coarse sampling time T_c, while Step 3 uses a finer sampling time T_f; and (*ii*) scope: Step 1 is solved for all possible input–output pairs, whereas Step 3 is solved by each satellite only for its assigned trajectory. Accordingly, in our first HIL testing we run the same TO problem at two sampling rates (coarse and fine) for a single initial–final position pair, since our focus is computation time; we set the coarse horizon to $T_c = 250$ and the fine horizon to $T_f = 20$. We also evaluate the TACA problem from Step 2 in our second HIL test. Because TACA is centralized and its complexity scales with formation size, we test different values of N to observe how the computation time grows with the number of satellites, specifically considering $N \in \{2, 10, 50\}$. In both HIL tests, the target orbit is assumed to be near-circular with semi-major axis $a = 6800$ km, eccentricity of 0.001, inclination $i = 98.6°$, right ascension of the ascending node $\Omega = 47°$, and argument of latitude, θ of 0°. The initial and desired formations for satellite i are $[\mathbf{x}_{\text{ini}}]_i = \text{col}\big(0,\ (i-1)\cdot 100\text{ m},\ 0,\ 0,\ 0,\ 0\big)$, and $[\mathbf{x}_{\text{des}}]_i = \text{col}\big(2000 + (i-1)\cdot 100\text{ m},\ 0,\ 0,\ 0,\ 0,\ 0\big)$.. For the TO problem, we set $u_{\max} = 1$ and $R_{\text{safe}} = 50$ m, with a horizon of one orbital period.

The goals of the HIL testing are to (*i*) verify that the TO, at both sampling rates, and the TACA, for all N, execute successfully on flight-class hardware, and (*ii*) quantify their computational demand for onboard feasibility. In our HIL setup, the TO and TACA were executed on a flight-class computer representative of a satellite, in collaboration with Benchmark Space Systems. The details of this setup are presented below.

Flight hardware and OS: Tests were performed on an Avnet MicroZed system-on-module featuring a Xilinx Zynq-7020 SoC (dual-core ARM Cortex-A9, 667 MHz) with 1 GB DDR3 RAM, running a lightweight Linux distribution built with PetaLinux. All algorithms were executed on the ARM processing system. The available interfaces during testing included RJ45 Ethernet, Ethernet-over-USB, and Wi-Fi.

Code base and solvers: The original MATLAB research code was ported to C++ for embedded deployment on flight hardware. All three steps of the trajectory planner require solving an optimization problem; therefore, we had to pick a solver capable of solving an LP for the TO problem in (6) and a MILP for the TACA problem in (12). Our desktop experiments used Gurobi for both optimization problems; however, Gurobi is not supported on the 32-bit ARM target. Accordingly, we pivoted to using Operator Splitting Quadratic Program (OSQP) to solve the TO problems and COIN-OR CBC to solve the TACA problem (see [81] and [82] for an introduction to OSQP and COIN-OR CBC).

Results: We solved TO for a single satellite pair ($i = 1$) on the flight hardware using OSQP with two discretizations, $T_f = 20$ and $T_c = 250$. We report solver time, wall time, iteration count on the flight CPU, and problem size in Table 2. As expected, increasing the sampling time reduces problem size and solve time. In fact, the solver time drops by almost $\times 10$ when increasing samoling time from 20 to 250s. The *solve time* refers to the time the solver spends computing a solution. The total *wall time* includes both the solve time and additional setup overhead, such as loading model data (e.g., $A(k)$ for $k = 1, \dots, K$ and B from (5)), initializing optimization parameters and initial conditions, and assembling the problem into the solver's required format (e.g., combining all equality and inequality constraints into a single matrix). On flight hardware, wall time exceeds solver time somewhat considerably due to uploading the time-varying dynamics matrices $A(k)$ and B (an effect that grows as the sampling time decreases); this transfer is absent in an actual in-flight run, where the data are already onboard.

After solving the TO with $T_c = 250$ s for all input–output pairs between $\mathbf{x}_{\text{ini}}$ and $\mathbf{x}_{\text{des}}$ for all $N \in \{2, 10, 50\}$, we populate the fuel matrix and solve the TACA problem. For each N, we report the solver time and wall time in Table 3. Note that here the wall time includes only uploading the fuel matrix and performing the matrix operations needed to match the solver's input format; we observe much smaller wall times, especially for smaller N, because the fuel matrix contains only a few values, in contrast to the large time-varying matrices handled by TO.

For the TACA, we need to consider the whole formation as the size of the formation directly affects the size of the porblem we use the same initial and final conditions as before and run. TACA solved for $N \in \{2,\ 10,\ 50\}$. For each configuration, we record solver time and end-to-end wall-time on the flight CPU. As expected both solver and wall times increase with N as the size of the problem increases. For $N = 50$, the solver takes about 12 seconds, which is is well below one minute, which we were aiming for. Note that TACA is not required to run in real time; it is invoked only at TP events. Moreover, because it is centralized, it need not run on flight hardware and can instead be executed on the ground or on a leader with higher computational power.

If we only consider the solver time, the TP cycle is dominated by the TACA solve rather than the TO. Combining the fuel–matrix calculation, TACA, and TO yields total solver times of about 0.55 s for $N = 2$, 0.58 s for $N = 10$, and 12.4 s for $N = 50$. These approximations are based solely on solver time; the corresponding wall times on the flight hardware will be larger, but not as large as the values reported in Table 2, because those measurements also include the overhead of uploading large data matrices to the microprocessor. Thus, if we are aiming for roughly one minute of computation per TP update, the computation-time trends in Table 3 and Table 2 suggest that the proposed control architecture is practically suitable for formations up to roughly $N \approx 50$ when solving TACA on the ground or with a dedicated leader.

Table 2 HIL results for Trajectory Optimization on flight hardware (OSQP).

Sampling time (s)	Problem Size		Flight Hardware (OSQP)	
	# Decision Vars (n)	# Constraints (m)	Solver (s)	Wall (s)
250	834	1887	0.0193	2.65
20	10230	23187	0.522	58.1

Table 3 HIL results for TACA (T_c = 250 s)

N	Solver Time (s)	Wall Time (s)
2	0.007435	0.0077
10	0.03787	0.0392
50	11.8483	12.1

VI. Conclusions

This work demonstrated a novel, integrated GNC framework that enables scalable, safe, and autonomous satellite swarm operations. The framework integrates (*i*) a formation planner that exploits swarm redundancy to improve communication-network reliability, (*ii*) a fuel-optimal, collision-aware trajectory planner that computes feasible reconfiguration maneuvers over a prescribed time horizon, and (*iii*) a control layer that couples real-time safety-critical control with robust state estimation to maintain performance under disturbances and sensor anomalies. High-fidelity simulations validated end-to-end performance, demonstrating formation reconfigurations while maintaining collision avoidance in the tested scenarios, including cases with previously unknown obstacles. Hardware-in-the-loop testing further showed that the trajectory planning algorithms are computationally feasible for execution on resource-constrained flight hardware.

While this work establishes a complete end-to-end GNC framework for autonomous satellite-swarm operations, several directions remain open for further research. First, the iterative formation planner warrants deeper theoretical analysis. Future work includes characterizing when the greedy placement converges to the globally optimal all-terminal reliability solution, deriving bounds on how suboptimal the solution can be in worst-case scenarios, and identifying conditions under which the greedy sequence is guaranteed to match the optimal placement. Second, the formation-planning problem can be broadened beyond communication-network reliability to include additional mission objectives, such as sensing coverage, geometric baselines for imaging, and load balancing, leading to multi-objective or jointly optimal terminal-formation designs. Third, reliability evaluation itself can be improved: in this work we estimate all-terminal reliability via Monte Carlo sampling, while a variety of methods for estimating and bounding network reliability have been proposed in the literature, including learning-based approaches that may offer improved run-time for formation planning in future work [83]. Fourth, for the TP, our proposed future work includes introducing curvature constraints in the decoupled TO problem to further improve fuel optimality, and studying the shrinking-horizon optimal control approach. Finally, larger-scale hardware-in-the-loop tests and eventual on-orbit demonstrations would further validate the computational scalability and practical feasibility of the proposed framework for next-generation swarm missions.

References

[1] Le Moigne, J., Little, M. M., and Cole, M. C., "New observing strategy (NOS) for future earth science missions," *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2019, pp. 5285–5288.

[2] Llorente, J. S., Agenjo, A., Carrascosa, C., de Negueruela, C., Mestreau-Garreau, A., Cropp, A., and Santovincenzo, A., "PROBA-3: Precise formation flying demonstration mission," *Acta Astronautica*, Vol. 82, No. 1, 2013, pp. 38–46.

[3] DeForest, C., Killough, R., Gibson, S., Henry, A., Case, T., Beasley, M., Laurent, G., Colaninno, R., and Waltham, N., "Polarimeter to unify the corona and heliosphere (punch): Science, status, and path to flight," *2022 IEEE Aerospace Conference (AERO)*, IEEE, 2022, pp. 1–11.

[4] Zink, M., Krieger, G., Fiedler, H., and Moreira, A., "The TanDEM-X mission: Overview and status," *2007 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2007, pp. 3944–3947.

[5] Ruf, C. S., Atlas, R., Chang, P. S., Clarizia, M. P., Garrison, J. L., Gleason, S., Katzberg, S. J., Jelenak, Z., Johnson, J. T., Majumdar, S. J., et al., "New ocean winds satellite mission to probe hurricanes and tropical convection," *Bulletin of the American Meteorological Society*, Vol. 97, No. 3, 2016, pp. 385–395.

[6] Zhihui, X., Jinguo, L., Chenchen, W., and Yuchuang, T., "Review of in-space assembly technologies," *Chinese Journal of Aeronautics*, Vol. 34, No. 11, 2021, pp. 21–47.

[7] Wulder, M. A., Loveland, T. R., Roy, D. P., Crawford, C. J., Masek, J. G., Woodcock, C. E., Allen, R. G., Anderson, M. C., Belward, A. S., Cohen, W. B., et al., "Current status of Landsat program, science, and applications," *Remote sensing of environment*, Vol. 225, 2019, pp. 127–147.

[8] Noh, M.-J., and Howat, I. M., "Analysis of planetscope dove digital surface model accuracy using geometrically simulated images," *Remote Sensing*, Vol. 15, No. 14, 2023, p. 3496.

[9] Di Mauro, G., Lawn, M., and Bevilacqua, R., "Survey on guidance navigation and control requirements for spacecraft formation-flying missions," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 3, 2018, pp. 581–602.

[10] Jain, R., Speretta, S., Dirkx, D., and Gill, E., "Inter-satellite tracking methods and applications: A comprehensive survey," *Advances in Space Research*, Vol. 74, No. 8, 2024, pp. 3877–3901.

[11] Paek, S. W., Kim, S., and de Weck, O., "Optimization of reconfigurable satellite constellations using simulated annealing and genetic algorithm," *Sensors*, Vol. 19, No. 4, 2019, p. 765.

[12] Leyva-Mayorga, I., Röper, M., Matthiesen, B., Dekorsy, A., Popovski, P., and Soret, B., "Inter-plane inter-satellite connectivity in LEO constellations: Beam switching vs. beam steering," *2021 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2021, pp. 1–6.

[13] Pinciroli, C., Birattari, M., Tuci, E., Dorigo, M., del Rey Zapatero, M., Vinko, T., and Izzo, D., "Self-organizing and scalable shape formation for a swarm of pico satellites," *2008 NASA/ESA Conference on Adaptive Hardware and Systems*, IEEE, 2008, pp. 57–61.

[14] Xu, Y., Zhang, Y., Wang, Z., He, Y., and Fan, L., "Self-organizing control of mega constellations for continuous Earth observation," *Remote Sensing*, Vol. 14, No. 22, 2022, p. 5896.

[15] Folta, D., Newman, L. K., and Quinn, D., "Design and implementation of satellite formations and constellations," *AAS/GSFC 13th International Symposium on Space Flight Dynamics*, Vol. 1, 1998.

[16] Boshuizen, C., Mason, J., Klupar, P., and Spanhake, S., "Results from the planet labs flock constellation," 2014.

[17] Kbidy, G., Adamski, G., and May, N., "Design concepts and challenges for the iridium NEXT command and control system," *2018 SpaceOps Conference*, 2018, p. 2708.

[18] Giralo, V., Chernick, M., and D'Amico, S., "Guidance, navigation, and control for the DWARF formation-flying mission," *AAS/AIAA Astrodynamics Specialist Conference, South Lake Tahoe, CA*, 2020, p. 0.

[19] Koenig, A. W., D'Amico, S., and Lightsey, E. G., "Formation flying orbit and control concept for virtual super optics reconfigurable swarm mission," *Journal of Guidance, Control, and Dynamics*, Vol. 46, No. 9, 2023, pp. 1657–1670.

[20] Bodin, P., Larsson, R., Nilsson, F., Chasset, C., Noteborn, R., and Nylund, M., "PRISMA: An in-orbit test bed for guidance, navigation, and control experiments," *Journal of Spacecraft and Rockets*, Vol. 46, No. 3, 2009, pp. 615–623.

[21] Oh, K.-K., Park, M.-C., and Ahn, H.-S., "A survey of multi-agent formation control," *Automatica*, Vol. 53, 2015, pp. 424–440.

[22] Ren, W., and Beard, R. W., *Distributed consensus in multi-vehicle cooperative control: theory and applications*, Springer, 2008.

[23] Mesbahi, M., and Egerstedt, M., *Graph Theoretic Methods in Multiagent Networks*, Princeton Series in Applied Mathematics, Princeton University Press, Princeton, NJ, 2010.

[24] Zavlanos, M. M., Egerstedt, M. B., and Pappas, G. J., "Graph-theoretic connectivity control of mobile robot networks," *Proceedings of the IEEE*, Vol. 99, No. 9, 2011, pp. 1525–1540.

[25] Colbourn, C. J., *The combinatorics of network reliability*, Oxford University Press, Inc., 1987.

[26] Morgan, D., Chung, S.-J., and Hadaegh, F. Y., "Model predictive control of swarms of spacecraft using sequential convex programming," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 1725–1740.

[27] Eren, U., Prach, A., Koçer, B. B., Raković, S. V., Kayacan, E., and Açıkmeşe, B., "Model predictive control in aerospace systems: Current state and opportunities," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 7, 2017, pp. 1541–1566.

[28] Das, P. K., and Jena, P. K., "Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators," *Applied Soft Computing*, Vol. 92, 2020, p. 106312.

[29] Olfati-Saber, R., Fax, J. A., and Murray, R. M., "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, Vol. 95, No. 1, 2007, pp. 215–233.

[30] Reynolds, C. W., "Flocks, herds and schools: A distributed behavioral model," *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987, pp. 25–34.

[31] Khatib, O., "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, Vol. 5, No. 1, 1986, pp. 90–98.

[32] Fax, J. A., and Murray, R. M., "Information flow and cooperative control of vehicle formations," *IEEE transactions on automatic control*, Vol. 49, No. 9, 2004, pp. 1465–1476.

[33] Yu, J., and LaValle, S. M., "Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics," *IEEE Transactions on Robotics*, Vol. 32, No. 5, 2016, pp. 1163–1177.

[34] Koren, Y., and Borenstein, J., "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation," *Proceedings of the 1991 IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 2, IEEE, IEEE, 1991, pp. 1398–1404.

[35] D'Amico, S., Gill, E., Garcia, M. F., and Montenbruck, O., "GPS-based real-time navigation for the PRISMA formation flying mission," *3rd ESA workshop on satellite navigation user equipment technologies, NAVITEC*, 2006.

[36] Zhang, Z., Deng, L., Feng, J., Chang, L., Li, D., and Qin, Y., "A survey of precision formation relative state measurement technology for distributed spacecraft," *Aerospace*, Vol. 9, No. 7, 2022, p. 362.

[37] Kahr, E., Roth, N., Montenbruck, O., Risi, B., and Zee, R. E., "GPS relative navigation for the CanX-4 and CanX-5 formation-flying nanosatellites," *Journal of Spacecraft and Rockets*, Vol. 55, No. 6, 2018, pp. 1545–1558.

[38] Montenbruck, O., Wermuth, M., and Kahle, R., "GPS based relative navigation for the TanDEM-X mission-first flight results," *Navigation*, Vol. 58, No. 4, 2011, pp. 293–304.

[39] Kaplan, E. D., and Hegarty, C., *Understanding GPS/GNSS: principles and applications*, Artech house, 2017.

[40] Priyadarshi, S., "A review of ionospheric scintillation models," *Surveys in geophysics*, Vol. 36, No. 2, 2015, pp. 295–324.

[41] McGraw, G. A., Groves, P. D., and Ashman, B. W., "Robust Positioning in the Presence of Multipath and NLOS GNSS Signals," *Position, Navigation, and Timing Technologies in the 21st Century: Integrated Satellite Navigation, Sensor Systems, and Civil Applications*, Vol. 1, Wiley, 2020, pp. 551–589.

[42] Psiaki, M. L., and Humphreys, T. E., "GNSS spoofing and detection," *Proceedings of the IEEE*, Vol. 104, No. 6, 2016, pp. 1258–1270.

[43] Gandhi, M. A., and Mili, L., "Robust Kalman filter based on a generalized maximum-likelihood-type estimator," *IEEE Transactions on Signal Processing*, Vol. 58, No. 5, 2009, pp. 2509–2520.

[44] Low, S. Y., Bell, T., and D'Amico, S., "Flight-Ready Precise and Robust Carrier-Phase GNSS Navigation Software for Distributed Space Systems," *arXiv preprint arXiv:2508.18246*, 2025.

[45] Lee, T., Leok, M., and McClamroch, N. H., "A combinatorial optimal control problem for spacecraft formation reconfiguration," *2007 46th IEEE Conference on Decision and Control*, IEEE, 2007, pp. 5370–5375.

[46] Chen, J., Wu, B., Sun, Z., and Wang, D., "Distributed safe trajectory optimization for large-scale spacecraft formation reconfiguration," *Acta Astronautica*, Vol. 214, 2024, pp. 125–136.

[47] Buchanan, C., Bagrow, J., Rombach, P., and Ossareh, H., "Node Placement to Maximize Reliability of a Communication Network with Application to Satellite Swarms," *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2023, pp. 3466–3473. https://doi.org/10.1109/SMC53992.2023.10394556.

[48] Basu, H., Pedari, Y., Almassalkhi, M., and Ossareh, H. R., "Computationally Efficient Collision-Free Trajectory Planning of Satellite Swarms Under Unmodeled Orbital Perturbations," *Journal of Guidance, Control, and Dynamics*, Vol. 46, No. 8, 2023, pp. 1548–1563. https://doi.org/10.2514/1.G007206.

[49] Pedari, Y., Basu, H., and Ossareh, H. R., "A Novel Framework for Trajectory Planning and Safe Navigation of Satellite Swarms," *IFAC-PapersOnLine*, Vol. 56, No. 3, 2023, pp. 547–552.

[50] Pedari, Y., Waleed, D., Espinosa, L. A. D., and Ossareh, H. R., "Robust State Estimation for Satellite Formations in the Presence of Unreliable Measurements," *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2023, pp. 2933–2939.

[51] Morgan, D., Chung, S.-J., Blackmore, L., Acikmese, B., Bayard, D., and Hadaegh, F. Y., "Swarm-keeping strategies for spacecraft under J_2 and atmospheric drag perturbations," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 5, 2012, pp. 1492–1506.

[52] Wu, B., Xu, G., and Cao, X., "Relative dynamics and control for satellite formation: accommodating J_2 perturbation," *Journal of Aerospace Engineering*, Vol. 29, No. 4, 2016, p. 04016011.

[53] Curtis, H. D., *Orbital mechanics for engineering students*, Butterworth-Heinemann, 2019.

[54] Schweighart, S. A., "Development and analysis of a high fidelity linearized J_2 model for satellite formation flying," Ph.D. thesis, Massachusetts Institute of Technology, 2001.

[55] Basu, H., Pedari, Y., Almassalkhi, M., and Ossareh, H. R., "Comparative Analysis of Satellite Relative Dynamics and Fuel-Optimal Trajectory Planning of Satellites Using Minimum Distance Assignment," *Proceedings of the 2022 American Control Conference (ACC)*, 2022, pp. 4023–4029. https://doi.org/10.23919/ACC53348.2022.9867860.

[56] Chen, Z., Zhang, H., Wang, X., Yang, J., and Dui, H., "Reliability analysis and redundancy design of satellite communication system based on a novel Bayesian environmental importance," *Reliability Engineering & System Safety*, Vol. 243, 2024, p. 109813.

[57] Agency, E. S., "About Space Debris," https://www.esa.int/Space_Safety/Space_Debris/About_space_debris, 2024. Accessed: January 16, 2024.

[58] Howell, E., "Van Allen Radiation Belts: Facts & Findings," https://www.space.com/33948-van-allen-radiation-belts.html, May 11 2018. Accessed: January 15 2024.

[59] "State-of-the-Art of Small Spacecraft Technology," https://www.nasa.gov/smallsat-institute/sst-soa/soa-communications/, Jul 28 2023. Accessed: January 15 2024.

[60] Dengiz, B., Altiparmak, F., and Smith, A. E., "Efficient optimization of all-terminal reliable networks, using an evolutionary approach," *IEEE transactions on Reliability*, Vol. 46, No. 1, 2002, pp. 18–26.

[61] Morgan, D., Chung, S.-J., Blackmore, L., Acikmese, B., Bayard, D., and Hadaegh, F. Y., "Swarm-Keeping Strategies for Spacecraft Under J2 and Atmospheric Drag Perturbations," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 5, 2012, pp. 1492–1506. https://doi.org/10.2514/1.55705, URL https://arc.aiaa.org/doi/10.2514/1.55705.

[62] AboElFotoh, H. M., and Colbourn, C. J., "Computing 2-terminal reliability for radio-broadcast networks," *IEEE Transactions on Reliability*, Vol. 38, No. 5, 2002, pp. 538–555.

[63] Paredes, R., Dueñas-Osorio, L., Meel, K. S., and Vardi, M. Y., "Principled network reliability approximation: A counting-based approach," *Reliability Engineering & System Safety*, Vol. 191, 2019, p. 106472.

[64] Yaglom, A. M., and Yaglom, I. M., *Challenging mathematical problems with elementary solutions*, Vol. I, Combinatorial Analysis and Probability Theory, Holden-Day, San Francisco, 1964. Translated by J. McCawley, Jr., revised and edited by B. Gordon.

[65] Welzl, E., "Smallest enclosing disks (balls and ellipsoids)," *New Results and New Trends in Computer Science*, Lecture Notes in Computer Science, Vol. 555, edited by H. Maurer, Springer, Berlin, Heidelberg, 1991, pp. 359–370.

[66] Ye, M., and Kolmanovsky, I., "Approximating optimal control by shrinking horizon model predictive control for spacecraft rendezvous and docking," *IFAC-PapersOnLine*, Vol. 55, No. 16, 2022, pp. 284–289. https://doi.org/10.1016/j.ifacol.2022.09.038, 18th IFAC Workshop on Control Applications of Optimization CAO 2022.

[67] Choset, H., "Robotic motion planning: Potential functions," *Robotics Institute, Carnegie Mellon University*, Vol. 164, 2010.

[68] Bounini, F., Gingras, D., Pollart, H., and Gruyer, D., “Modified artificial potential field method for online path planning applications,” *2017 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 180–185.

[69] Park, M. G., Jeon, J. H., and Lee, M. C., “Obstacle Avoidance for Mobile Robots Using Artificial Potential Field Approach with Simulated Annealing,” *Proceedings of the 2001 IEEE International Symposium on Industrial Electronics (ISIE)*, Vol. 3, IEEE, Pusan, South Korea, 2001, pp. 1530–1535. https://doi.org/10.1109/ISIE.2001.931933, cat. No. 01TH8570.

[70] Bodin, P., Noteborn, R., Larsson, R., Karlsson, T., D’Amico, S., Ardaens, J. S., Delpech, M., and Berges, J.-C., “The prisma formation flying demonstrator: Overview and conclusions from the nominal mission,” *Advances in the Astronautical Sciences*, Vol. 144, No. 2012, 2012, pp. 441–460.

[71] Roth, N. H., Risi, B., Grant, C., and Zee, R., “Flight results from the CanX-4 and CanX-5 formation flying mission,” *Proceedings of the Small Satellites, Systems & Services Symposium (4S), Valletta, Malta*, 2016, pp. 1–15.

[72] Schaub, H., and Junkins, J. L., *Analytical mechanics of space systems*, Aiaa, 2003.

[73] Gao, Z., Fang, K., Wang, Z., Guo, K., and Liu, Y., “An error overbounding method based on a Gaussian mixture model with uncertainty estimation for a dual-frequency ground-based augmentation system,” *Remote Sensing*, Vol. 14, No. 5, 2022, p. 1111.

[74] Pfeifer, T., and Protzel, P., “Expectation-maximization for adaptive mixture models in graph optimization,” *2019 international conference on robotics and automation (ICRA)*, IEEE, 2019, pp. 3151–3157.

[75] Gupta, S., and Gao, G., “Data-driven protection levels for camera and 3D map-based safe urban localization,” *Navigation*, Vol. 68, No. 3, 2021, pp. 643–660.

[76] NovAtel Inc., “OEMV-1 Series Product Sheet (incl. OEMV-1G),” https://hexagondownloads.blob.core.windows.net/public/Novatel/assets/Documents/Papers/OEMV-1_Series/OEMV-1_Series.pdf, 2013. D14938 Rev 3. Single-point L1 horizontal accuracy: 1.5 m RMS.

[77] Howard, R. T., Heaton, A. F., Pinson, R. M., and Carrington, C. K., “Orbital express advanced video guidance sensor,” *2008 IEEE Aerospace Conference*, IEEE, 2008, pp. 1–10.

[78] Aglietti, G. S., Taylor, B., Fellowes, S., Ainley, S., Tye, D., Cox, C., Zarkesh, A., Mafficini, A., Vinkoff, N., Bashford, K., et al., “RemoveDEBRIS: An in-orbit demonstration of technologies for the removal of space debris,” *The Aeronautical Journal*, Vol. 124, No. 1271, 2020, pp. 1–23.

[79] Ruel, S., Luu, T., and Berube, A., “Space shuttle testing of the TriDAR 3D rendezvous and docking sensor,” *Journal of Field robotics*, Vol. 29, No. 4, 2012, pp. 535–553.

[80] Delpech, M., Guidotti, P.-Y., Grelier, T., and Harr, J., “RF based navigation for PRISMA and other formation flying missions in earth orbits,” *Advances in the Astronautical Sciences*, Vol. 135, No. 2, 2009, pp. 1533–1551.

[81] Stellato, B., Banjac, G., Goulart, P., Bemporad, A., and Boyd, S., “OSQP: An operator splitting solver for quadratic programs,” *Mathematical Programming Computation*, Vol. 12, No. 4, 2020, pp. 637–672.

[82] Forrest, J., and Lougee-Heimer, R., “CBC user guide,” *Emerging theory, methods, and applications*, INFORMS, 2005, pp. 257–277.

[83] Srivaree-Ratana, C., Konak, A., and Smith, A. E., “Estimation of all-terminal network reliability using an artificial neural network,” *Computers & Operations Research*, Vol. 29, No. 7, 2002, pp. 849–868.