2017

Dynamic air-bearing hardware-in-the-loop testbed to experimentally evaluate autonomous spacecraft proximity maneuvers

Zappulla, Richard II; Virgili-Llop, Josep; Zagaris, Costantinos; Park, Hyeongjun; Romano, Marcello

AIAA

R. Zappulla II, J. Virgili-Llop, C. Zagaris, H. Park, M. Romano, "Dynamic sir-bearing hardware-in-the-loop testbed to experimentally evaluate autonomous spacecraft proximity maneuvers," Journal of Spacecraft and Rockets, (Article in Advance) (2017),15 p.

http://hdl.handle.net/10945/53759

Dynamic Air-Bearing Hardware-in-the-Loop Testbed to Experimentally Evaluate Autonomous Spacecraft Proximity Maneuvers

Richard Zappulla, II,* Josep Virgili-Llop,† Costantinos Zagaris,‡ Hyeongjun Park,§ and Marcello Romano¶
Naval Postgraduate School, Monterey, California 93943

DOI: 10.2514/1.A33769

Ground-based testbeds are critical to develop and test different elements of spacecraft guidance, navigation, and control subsystems. This paper provides an in-detail description of a state-of-the-art air-bearing testbed used to develop guidance, navigation, and control methods for close-proximity operations. Test vehicles, representing spacecraft, float via air bearings over a horizontally leveled granite monolith and move with two translational degrees of freedom and one rotational degree of freedom under the effect of thrusters and reaction wheel actuators. This setup achieves a quasi-frictionless and low residual acceleration dynamic environment. The testbed experimental setup as well as the vehicle hardware and software architectures are discussed in detail. Characterization of different testbed elements is provided. Finally, a test campaign is used to showcase its capabilities and to illustrate the testbed operations.

Nomenclature

a_n	=	curve fit coefficient
$\boldsymbol{B}$	=	continuous-time control input matrix
F	=	force, N
$\boldsymbol{f}$	=	vector of forces applied by each thruster
$\boldsymbol{H}_k$	=	measurement mapping matrix at sample time k
$\boldsymbol{I}$	=	identity matrix
I_{RW}	=	reaction wheel inertia, kg · m^2
J_{zz}	=	moment of inertia about the z axis, kg · m^2
$\boldsymbol{K}_a$	=	attractive potential gain matrix
$\boldsymbol{K}_k$	=	Kalman gain at sample time k
k	=	sample-time index
$\boldsymbol{M}$	=	thruster to control input mapping matrix
m	=	mass, kg
$\boldsymbol{P}_k$	=	error covariance at sample time k
$\boldsymbol{Q}_a$	=	attractive potential shaping matrix
$\boldsymbol{Q}_k$	=	process noise covariance at sample time k
q_k	=	scalar quantity for Q_k at sample time k
$\boldsymbol{R}_k$	=	sensor noise covariance at sample time k
r_{ref}	=	circular reference trajectory radius, m
t	=	time, s
$\boldsymbol{u}_k$	=	control input vector at sample time k
x, y	=	position, m
$\boldsymbol{x}_c$	=	center of circular trajectory vector
$\boldsymbol{x}_k$	=	state vector at sample time k
$\boldsymbol{x}_{\mathrm{ref}}$	=	reference position vector
$\hat{\boldsymbol{x}}_k$	=	state estimate at sample time k
$\dot{x}$, $\dot{y}$	=	velocity, m/s
$\boldsymbol{z}_k$	=	measurement vector at sample time k
α	=	reaction wheel acceleration, rad/s^2
β	=	angle between floating spacecraft simulator and point tangent to reference trajectory, rad
$\boldsymbol{\Gamma}_k$	=	discrete-time input gain matrix at sample time k
Δt_{DKF}	=	discrete Kalman filter time step, s
ϵ_{DKF}	=	discrete Kalman filter convergence threshold
ζ	=	χ^2 test statistic for measurement association
$\boldsymbol{\Upsilon}_\kappa$	=	process noise covariance matrix at sample time k
θ	=	angle, rad
$\dot{\theta}$	=	angular rate, rad/s
σ_x, σ_y	=	position noise, m
σ_θ	=	attitude sensor noise, rad
σ_ω	=	fiber-optic gyroscope sensor noise, rad/s
τ	=	torque, N · m
$\boldsymbol{\Phi}_k$	=	state transition matrix at sample time k
ϕ	=	angle of floating spacecraft simulator relative to center of the reference trajectory, rad
$\dot{\psi}_{\mathrm{ref}}$	=	user-specified desired angular rate, rad/s
ω_k	=	process noise at sample time k

Presented as Paper 2016-5268 at the AIAA/AAS Astrodynamics Specialist Conference 2016, Long Beach, CA, 12–16 September 2016; received 21 September 2016; revision received 11 January 2017; accepted for publication 22 January 2017; published online 11 April 2017.

*Ph.D. Candidate, Mechanical and Aerospace Engineering Department, 1 University Circle. Student Member AIAA.
†Postdoctoral Research Associate, Mechanical and Aerospace Engineering Department, 1 University Circle. Member AIAA.
‡Ph.D. Student, Mechanical and Aerospace Engineering Department, 1 University Circle.
§Postdoctoral Research Associate, Mechanical and Aerospace Engineering Department, 1 University Circle. Member AIAA.
¶Associate Professor, Mechanical and Aerospace Engineering Department, 1 University Circle. Associate Fellow AIAA.

I. Introduction

ADVANCEMENTS in autonomous navigation systems and onboard guidance, navigation, and control (GNC) are deemed to be essential in order to pursue the wide variety of future space missions identified in a National Research Council's decadal study entitled "Voyages for Planetary Science in the Decade 2013–2022" [1]. The 2015 National Aeronautics and Space Administration (NASA) report on technology roadmaps also identified the need for advancements in system-level autonomy and on-orbit robotics, as well as in autonomous rendezvous and docking, in order to support a wide range of envisioned space missions [2]. Both studies emphasized the need to improve, among others, the onboard sensing and filtering capabilities; the ability to autonomously detect and react to events or in situ disturbances; as well as the ability to improve onboard autonomous trajectory optimization, path planning, and replanning capabilities. The 2015 NASA report on technology roadmaps additionally stated integration, verification, and validation were some of the greatest challenges that future autonomous GNC systems faced. To achieve this, Quadrelli et al. recommended investing into "system-level demonstration systems, such as ground-based end-to-end GNC system testbeds ..." in order to ensure GNC performance requirements were being

achieved [1]. In addition to allowing for more complex scientific planetary missions, these advancements would also enable safer and more capable autonomous rendezvous and proximity operations (RPO).

Over the past decade, numerous missions have been proposed and attempted, exploring various techniques for safely conducting RPOs: the U.S. Air Force Research Laboratory's XSS-10 and XSS-11 missions [3,4]; NASA's Demonstration of Autonomous Rendezvous Technology (known as DART) mission [5]; the Defense Advanced Research Projects Agency's Orbital Express, Spacecraft for the Universal Modification of Orbits/Front-End Robotics Enabling Near-term Demonstration (known as SUMO/FREND), and Phoenix missions [6–8]; the Swedish Space Corporation's PRISMA mission [9]; the University of Texas at Austin's Low Earth Orbiting Navigation Experiment for Spacecraft Testing Autonomous Rendezvous and Docking (known as LONESTAR) Bevo-2 mission [10]; and lastly, the Georgia Institute of Technology's Prox-1 mission [11,12]. To ensure safe on-orbit operations, ground testing must be performed to ensure that the resulting GNC algorithms meet a predefined set of performance requirements. To perform this task, at the recommendation of Quadrelli et al., it is argued that ground-based system-level demonstration systems can provide adequate fidelity to verify and validate GNC performance requirements [1].

Air-bearing testbeds provide a dynamically representative environment to develop and test GNC algorithms. In these facilities, the test vehicles that represent spacecraft (or resident space objects) operate on top of a planar surface. Air-bearings located on the test vehicles are used to reduce the friction of the vehicles to create a quasi-frictionless environment. As the planar surface is horizontally leveled, the effects of any in-plane components of gravity on the test vehicles are minimal. The result is a quasi-frictionless and low residual acceleration environment in a plane, thus emulating the drag-free and weightless environment of orbital spaceflight. Additionally, these testbeds include various hardware phenomena which are practically impossible to replicate in a simulated environment. These phenomena include: delays, computational constraints, actuator response uncertainty, and sensor noise. Ground-based GNC testbeds are a useful tool to advance the state of the art of these systems and can be used to perform end-to-end system-level verification and validation before the system's operational deployment.

A. Survey of Relevant Testbeds

Although survey papers on air-bearing spacecraft simulators for RPO exist in the literature [13,14], new facilities have emerged since these were published. A summary of the characteristics of the examined testbeds not found in the survey literature is tabulated in Table 1.

Both the University of Florida's (UF's) ADvanced Autonomous MUltiple Spacecraft testbed and the Florida Institute of Technology's (FIT's) Orbital Robotic Interaction, On-orbit servicing, and Navigation (with its Diverse Air-beaing Weightless eNvironment vehicle) testbed are capable of full six-degree-of-freedom (6-DOF) motion [15–17]. In these testbeds, planar air bearings are used by the test vehicles to translate on top of an epoxy floor, while a spherical air-bearing provides the rotational 3-DOF, and a complex counterbalance system is used to provide frictionless out-of-plane motion. The Georgia Institute of Technology's Autonomous Spacecraft Testing of Robotic Operations in Space (ASTROS) uses a 5-DOF platform, where the ASTROS platform is a 2-DOF rotational air bearing atop a 3-DOF platform that employs translation air bearings [18]. Lastly, the University of Rome "La Sapienza" Platform Integrating Navigation and Orbital Control Capabilities Hosting Intelligence Onboard (PINOCCHIO) testbed is a two-translational DOF and one-rotational DOF testbed using only a translational planar air bearing [19].

With the exception of the PINOCCHIO testbed at the University of Rome "La Sapienza", which is still in development, all the examined testbeds use a commercial motion capture system, such as Vicon or PhaseSpace, in order to provide an inertial position to the spacecraft simulator. These external position measurements are usually augmented by an onboard inertial measurement unit (IMU) and fused together for more accurate state estimations.

Furthermore, the use of an epoxy surface is pervasive throughout the surveyed air-bearing testbeds. Although granite surfaces provide unmatched surface planarity and smoothness, epoxy floors or glass surfaces have a significantly lower procurement cost, thus making them more common. Lastly, it is worthwhile to note that a detailed characterization of the testbeds is typically not found in the literature, thus hindering the comparison of the experimental results obtained when using different testbeds. From an experimental standpoint, having such a detailed description and characterization is important in order to understand the system's performance (e.g., navigation performance) and uncertainties (e.g., actuator performance, moment of inertia estimate, and residual accelerations) under which a GNC algorithm has been experimentally evaluated.

B. Naval Postgraduate School POSEIDYN Testbed

In this paper, a detailed description of the Naval Postgraduate School (NPS) Proximity Operation of Spacecraft: Experimental hardware-In-the-loop DYNamic simulator (POSEIDYN) air-bearing testbed is provided. The POSEIDYN testbed has been developed with the aim of providing a representative system-level platform upon which to develop, experimentally test, and partially validate GNC algorithms for RPO. Over the decade that the testbed has been in operation and in constant upgrade, four different generations of floating spacecraft simulators (FSSs) have been developed. The evolution of these test vehicles is shown in Fig. 1. Each FSS generation has included a unique capability. The first-generation FSSs used a prototype capture system in order to perform rendezvous and docking [20]. The second-generation FSSs featured vectorable thrusters and a miniature control moment gyroscope [21,22]. The third-generation FSSs moved away from an aluminum construction in favor of a more lightweight, polycarbonate structure (fabricated using additive manufacturing) and components, such as docking interfaces. Lastly, the fourth-generation FSSs continued the use of polycarbonate components and included a standardized interface for use with robotic manipulator research [23].

Before the installation of the 4×4 m granite monolith in early 2012, the vehicles operated over an epoxy floor. An overview of the testbed is shown in Fig. 2.

For examples of recent research efforts performed using this testbed, the reader is encouraged to read [20,23–35].

Table 1 Summary of the relevant testbeds characteristics [15–19]

Name	Location	Simulator type	Surface material	Navigation sensors
ADAMUS	UF	Translational + Rotational Air-Bearing	Epoxy	PhaseSpace
ASTROS	Georgia Institute of Technology	Translational + Rotational Air-Bearing	Epoxy	Vicon SICK laser sensor
ORION	FIT	Translational + Rotational Air-Bearing	Epoxy	IMU OptiTrack
PINOCCHIO	University of Rome "La Sapienza"	Translational Air-Bearing	Glass	Onboard IMU
POSEIDYN	NPS	Translational Air-Bearing	Granite	Vicon FOG

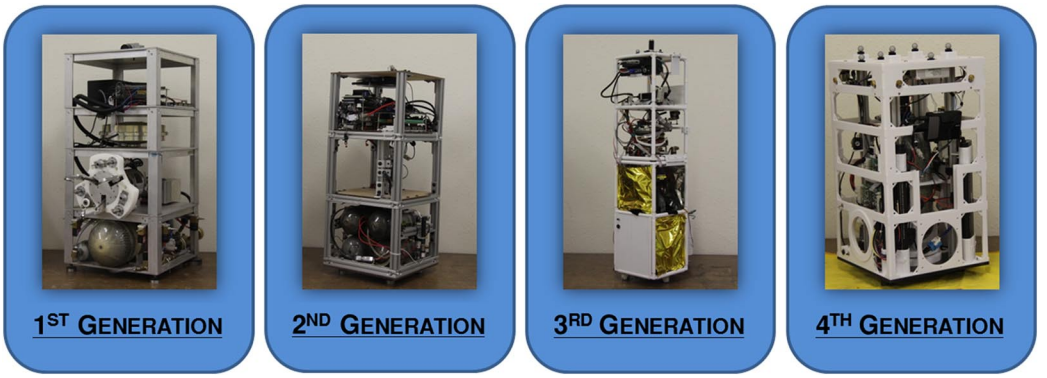


Fig. 1 Lineup of the first- to fourth-generation floating spacecraft simulators used on the POSEIDYN testbed.

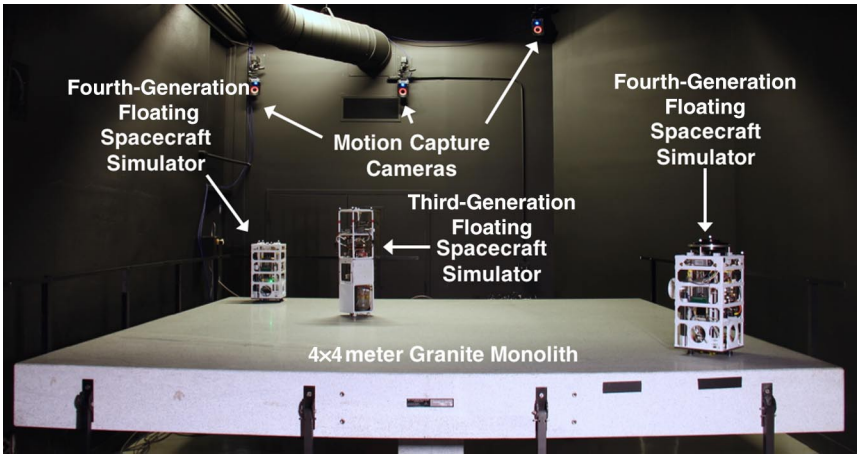


Fig. 2 Overview of the main elements of the POSEIDYN testbed.

C. Contributions and Layout

In this paper, a detailed description will be given of the hardware and software architectures of the POSEIDYN testbed, including a thorough characterization of the system. Furthermore, a demonstration of the capabilities of the POSEIDYN testbed for system-level development and experimental evaluation of GNC algorithms will be reported.

The layout of the remainder of the paper is as follows. First, both the hardware and software architectures of the POSEIDYN testbed are described in Sec. II. Next, the operating system (OS) latencies, sensor noises, thruster performance, physical properties, and end-to-end residual accelerations are characterized in Sec. III. In Sec. IV, a case study (where the FSS enters into a circular trajectory around a fixed point and follows it at a constant angular velocity for four revolutions) is used in order to demonstrate the capabilities of the testbed. Lastly, concluding remarks are drawn.

II. Hardware and Software Architecture

A. Hardware Architecture

The POSEIDYN testbed is composed of three main elements: a 4×4 m granite monolith, multiple FSSs, and a laboratory-wide metrology system. An overview of these elements is shown in Fig. 2.

The FSSs are custom-designed vehicles that emulate orbital spacecraft moving in close proximity of another vehicle or object (e.g., another FSS). In this paper, the fourth-generation FSSs will be considered. Three flat 25-mm-diam air-bearings are used by the FSSs to achieve quasi-frictionless motion on top of the granite monolith. The air-bearings use compressed air to lift the FSSs approximately 5 μm, creating an air film between the vehicle and the granite surface that acts as a lubricant layer and eliminates their direct contact. The compressed air, supplied at 60 psi (4.1 bar), is delivered from an onboard tank holding 1.87 liters of compressed air at 3000 psi (206.8 bar). With a nominal air consumption of 0.1 l/min at 60 psi (0.53 normalized liters per minute) per air-bearing, the floating endurance of the FSS is estimated to be at approximately 3.5 h. A solenoid valve controls the flow toward the air-bearings, and an air filter prevents contaminants and foreign materials from damaging the delicate air-bearing porous material. A scuba-diving-type compressor is used to refill the onboard tank.

The 15 metric ton, 4×4 m granite monolith is supported by three adjustable pedestals and has a planar accuracy of ± 0.0127 mm. This allows the granite surface to be horizontally leveled to an accuracy of 0.01 deg. Seismic activity and other building structural activity can distort the monolith alignment over time. Periodic checks are performed to ensure that the granite monolith is leveled to within the prespecified tolerance.

The combination of the reduced friction (provided by the air-bearings) with the low residual acceleration (provided by the horizontality and planarity of the granite monolith) allows the POSEIDYN testbed to recreate the drag-free and weightless environment experienced by spacecraft in close-proximity maneuvering. A characterization of the residual acceleration experienced by the FSS can be found in Sec. III. An important limitation of the testbed consists of allowing only 3-DOF motion (two translation and one rotational), as compared to 6-DOF motion of a spacecraft on orbit. Furthermore, the applicability of the testbed, as a high-fidelity dynamic simulator, is limited to short-duration close-proximity operations because the relative orbital mechanics are not recreated. Despite these limitations, the POSEIDYN testbed is critical to experimentally test guidance, navigation, and control methods for proximity operations in a realistic environment, including sensor and actuator dynamics, as well as real-time computational constraints.

To propel the FSS over the granite monolith, the vehicles are equipped with eight cold-gas thrusters, with two thrusters mounted equidistant from the center of the FSS. Each thruster is composed of a solenoid valve and a custom-made convergent–divergent nozzle [36]. Fed by the onboard tank of compressed air, the thrusters provide a nominal thrust between 0.1 and 0.15 N of thrust (see Sec. III for thruster characterization details). This nominal thrust fluctuates

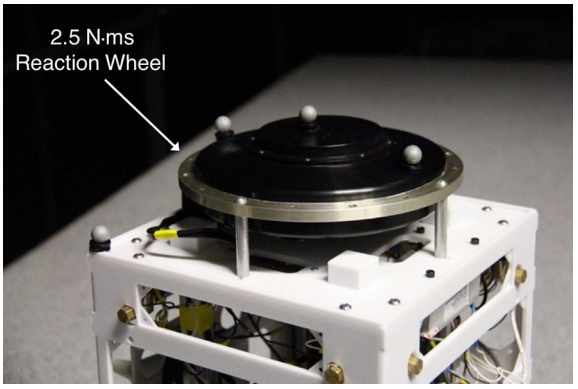


Fig. 3 Ball Aerospace 2.5 N · m · s reaction wheel mounted atop one of the FSSs.

considerably, as the thrust is a function of nozzle inlet pressure, which changes depending on the number of thrusters that are being fired simultaneously. With a nominal mass flow of about 0.3 g/s per thruster, the total firing time is estimated to be 20 min. Red light-emitting diodes paired with each thruster provide a visual indication of which thruster is firing.

Furthermore, one of the fourth-generation FSSs has a Ball Aerospace 2.5 N · m · s reaction wheel (RW) (model RW 2.5-A1) mounted atop its structure, as illustrated in Fig. 3. This particular FSS is used for spacecraft robotics research, and a multilink modular robotic manipulator can be connected to it [23]. The RW was installed to meet the increased torque requirements due to the dynamic coupling (arising from the manipulator motion) and for the significantly higher inertia of the combined system. An Arduino Due microcontroller is used as an interface between the onboard computer and the RW. This microcontroller converts the output of the three hall sensors on the RW to a rotation rate and provides a suitable analog signal to control the RW motor torque. A switching shunt regulator is used to safely dissipate the energy generated during RW deceleration.

The onboard computational capabilities of the FSS are provided by a commercial-off-the-shelf PC-104 form-factor onboard computer. Based on an Intel Atom 1.6 GHz, 32-bit processor, the computer has 2 GB of RAM and an 8 GB solid-state drive. Despite the onboard computer not being a space-qualified compute system (such as the BAE RAD750), its computational capabilities could be regarded to be on the same order of magnitude as state-of-the-art space-grade computers, as illustrated in Table 2 [37–43]. An expansion board with 20 optoisolated relays provides the required switching capability for the thrusters and air-bearing solenoid valves. A serial interface is used to communicate with an onboard KVH DSP-3000 fiber-optic gyroscope (FOG), which provides angular velocity measurements at a 100 Hz rate. On the FSS with the RW, another serial port is used to communicate with the microcontroller interfacing with the RW. Two 95 W · h lithium–ion batteries and a battery management module regulate the electrical power to the FSS. Under idle conditions, the batteries can power the FSS for over 5.5 h; under maximum loading, including the RW, the FSS can operate for just under 3.5 h, thus making the amount of air inside the onboard tank the limiting factor during experiments.

A Wi-Fi module provides the FSS with wireless communication capabilities [44]. The Wi-Fi module enables the FSS to communicate with other FSSs or other external computers, such as the Vicon server and ground station for telemetry logging and visualization purposes. To minimize latency, the data are transmitted from node to node using the user datagram protocol (UDP).

A carbon-fiber-reinforced polymer base plate and four aluminum T-slotted columns make up the core of the FSS structure. A polycarbonate outer shell provides the attachment points for all the FSS components. Fabricated using additive manufacturing, the outer shell can be quickly modified to accommodate any component changes or vehicle upgrades (e.g., this was done with the RW).

The laboratory motion-capture system is a commercial system (Vicon) composed of 10 overhead cameras and an external computer. This system determines the position of objects carrying passive markers (i.e., the FSS) with submillimeter-level (static) accuracy at rates up to 100 Hz. Once the location of the FSS is determined by the Vicon system, an external computer streams the data to the FSS using the Wi-Fi link.

B. Software Architecture

At the core of the FSS software architecture is a real-time (RT) OS, which ensures the overlaying GNC software running on board responds to sensor inputs and generates the appropriate actuator outputs within a strict and predefined time span. To achieve the desired real-time requirement, a Ubuntu 10.04, 32-bit server-edition operating system has been chosen, and its Linux kernel (Version 2.6.33) has been patched with the RT-Preempt patch [45,46]. This particular OS combination used by the FSS will be referred as RT-Linux OS. Running atop the RT-Linux OS is the GNC software, which can be broken up into four main subsystems: navigation, guidance, control, and telemetry.

The navigation subsystem provides a full state estimate of the FSS (position, orientation, and rates). Because most of the research conducted on the POSEIDYN testbed is focused on the guidance and control aspect, the navigation subsystem makes use of the accurate Vicon information to provide an accurate state estimate. As illustrated in Fig. 4, the navigation block first samples the onboard sensors and the actuator states, and it fuses them via a discrete Kalman filter (DKF) to produce an inertial state estimate. The inertial state estimate is then sent to the guidance subsystem, where the appropriate actuator commands are generated and sent to the control subsystem. The standard control subsystem is comprised of the steering logic and converts the desired actuator inputs to the required low-level signals to drive the different onboard actuators. Typically, only the guidance subsystem is user-defined function, and the navigation and control subsystems use predefined software. However, when research is performed on relative navigation, for example, the navigation block can be used to provide a "ground truth" estimate to benchmark the experimental relative navigation results [20].

Lastly, the telemetry subsystem packages the requested telemetry and sends it to a desired ground station. In addition to the GNC telemetry specified in the model, a publicly available system monitoring utility is used to collect metrics for various system-level

Table 2 Comparison of space-qualified and typical terrestrial processors [37–43] (MIPS, millions of instructions per second)

	Name	Number of cores	Number of threads	Processor frequency, MHz	Peak power, W	Estimated MIPS	MIPS per watt
Space-qualified processors	BAE RAD750	1	1	200	12.5	400	32
	Proton200k DSP	1	1	200	5	4,000	800
	Proton200k Lite	1	1	66	1.5	1,200	800
	Tyvak Intrepid (ARM processor)	1	1	400	0.3	440	1,466.67
Terrestrial processors	ARM Cortex A15 (Tegra processor)	4	4	1,900	2.25	4,000	1,777.78
	Intel Core i7-4710MQ (generic laptop CPU)	4	8	2,500	47	40,000	851.06
	Intel Atom processor (FSS processor)	1	2	1,600	5	3,200	640

metrics such as, but not limited to, processor usage, memory consumption, and network bandwidth. In particular, the `sysstat` utility was chosen [47]. It is worthwhile to note that the sampling rate of each subsystem (navigation, guidance, and control) is user specified, allowing for a multirate GNC formulation.

The remainder of this section will detail specifics about the GNC software development, navigation formulation, thruster mapping and modulation, reaction wheel control, and the development simulator.

1. GNC Software Development

To simplify the algorithm development and subsequent implementation on the FSS, a numerical development simulator and a FSS software template were created using a common custom library. This library contains common software (navigation and control subsystems), which is used in both the simulator and the FSS autogenerated onboard software. The simulator uses simulated sensors and actuators, and it simulates the plant (i.e., FSS) response; whereas the FSS software template uses the interfaces to the onboard sensors and actuators. This commonality between the simulator and FSS onboard software allows for rapid development of the algorithms in a simulation environment and software generation for use onboard the FSS for testing.

The multirate GNC software running atop the RT-Linux OS is developed using the MATLAB and Simulink R2015b environment. Once developed, the Simulink models are autocoded into C and compiled (using the `ert_linux` target language compiler [48]). To facilitate the code generation and compilation across multiple OSs and architectures, a Ubuntu virtual machine (VM) has been created and the development tool chain has been preinstalled to reduce the VM setup. This VM allows the user to easily create hard real-time multirate GNC software.

2. Navigation Subsystem

A DKF fuses the Vicon and FOG data as well as an estimate of the actuated force and torque (given the states of the thruster valves and RW torque) to provide an estimate of the FSS inertial state. The FSS DKF is conceptually broken up into an outer-loop consisting of the prediction steps and an inner-loop composed of the correction steps that update the outer-loop only when the filter is converged. Furthermore, in order to retain flexibility, the DKF operates at a user-defined time step Δt_{DKF}. However, dropped or corrupted sensor measurements are inevitable due to the asynchronous sensors, user-defined time step, and lack of error correction in UDP communications. A dynamic construction of the measurement mapping matrix is used to mitigate the effect of dropped sensors measurement, and a χ^2-gating rejects corrupted out-of-bound measurements [49]. Lastly, in addition to the standard output of a state estimate and error covariance estimate, additional telemetry indicating the status and "health" of the filter (e.g., initialization status, convergence status, measurement validity, and measurement gating) is stored in the bit field of a 32 bit integer. This section will detail the construction and considerations given in the development of the FSS DKF.

To formulate the DKF, the system is assumed to take the discrete-time representation shown in Eq. (1). The state vector is denoted by $\boldsymbol{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k, \theta, \dot{\theta}_k]^T$; the system state transition matrix by Φ_k and ω_k denotes the process noise assumed to be zero mean Guassian white noise at time step k:

$$\boldsymbol{x}_k = \Phi_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{\Gamma}_{k-1}\boldsymbol{u}_{k-1} + \boldsymbol{\Upsilon}_{k-1}\omega_{k-1} \tag{1}$$

$\boldsymbol{\Gamma}_k$ is the discrete-time input gain matrix at sample time k; and $\boldsymbol{\Upsilon}_\kappa$ is the process noise covariance matrix at sample time k.

The predicted state estimate and predicted error covariance are given as follows:

$$\hat{\boldsymbol{x}}_{k|k-1} = \Phi_k\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{\Gamma}_k\boldsymbol{u}_{k-1} \tag{2}$$

$$\boldsymbol{P}_{k|k-1} = \Phi_k\boldsymbol{P}_{k|k-1}\Phi_k^T + \boldsymbol{Q}_k \tag{3}$$

where $\boldsymbol{u}_{k-1} = [F_x, F_y, \tau]^T$ denotes the control input at the previous time step, and the process noise covariance $\boldsymbol{Q}_k \in \mathbb{R}^{6\times6}$ used in the propagation of the predicted error covariance is assumed to have the following form [49,50]:

$$\boldsymbol{Q}_k = q_k^2\begin{bmatrix} \Delta t_{\text{DKF}}^4/4 & \Delta t_{\text{DKF}}^3/2 & 0 & 0 & 0 & 0 \\ \Delta t_{\text{DKF}}^3/2 & \Delta t_{\text{DKF}}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \Delta t_{\text{DKF}}^4/4 & \Delta t_{\text{DKF}}^3/2 & 0 & 0 \\ 0 & 0 & \Delta t_{\text{DKF}}^3/2 & \Delta t_{\text{DKF}}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Delta t_{\text{DKF}}^4/4 & \Delta t_{\text{DKF}}^3/2 \\ 0 & 0 & 0 & 0 & \Delta t_{\text{DKF}}^3/2 & \Delta t_{\text{DKF}}^2 \end{bmatrix} \tag{4}$$

For simplicity, it is assumed the output of the thrusters follow a square wave; that is, the output of the thrusters is modeled to be fully on or fully off with no transient response. To generalize the implementation of the filter, a forward initialization method is used to provide an in situ initial state estimate and initial error covariance using two complete measurement sets [49]. Furthermore, to compensate for uncertainties associated with thrusting (see Sec. III), the noise intensity q_k^2 associated with the attitude rate process noise diagonal element see Eq. (4) is increased by a factor of 500 from a nominal value of 1×10^{-3}. This is done because the attitude rate, due to the small moment of inertia of the FSS, is more sensitive to thrust uncertainties.

The next consideration in the construction of the FSS DKF is the dynamic development of the measurement mapping matrix $\boldsymbol{H}_k \in \mathbb{R}^{4\times6}$. Nominally, the measurement mapping matrix is defined as follows:

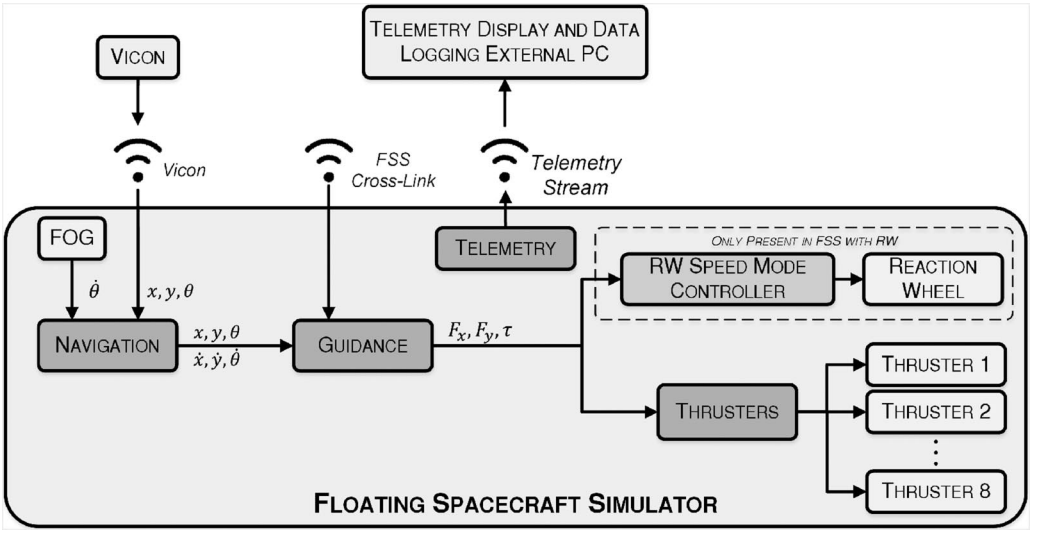


Fig. 4 Overview of the onboard software architecture of the floating spacecraft simulators.

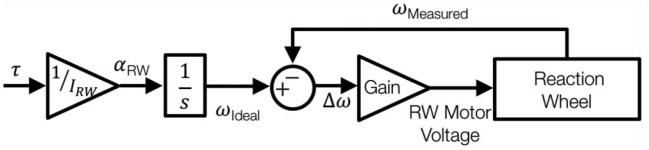


Fig. 5 Reaction wheel speed-mode controller.

$$\boldsymbol{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

The first three rows are associated with measurements from the Vicon sensor, whereas the last row is associated with measurements from the FOG. The dynamic construction of the measurement mapping matrix involves ensuring each measurement received from each sensor is "valid" before it is processed. The data validity check is a two-step process; first, the `Data_Available` flag from the Vicon sensor and FOG are checked to be `true`. If either (or both) `Data_Available` flags are `false`, the respective rows in the nominal measurement mapping matrix $\boldsymbol{H}_k$ in Eq. (5) are set to a row of zeros and the respective bit in the DKF health telemetry item is set to "0." It is important to note that the `Data_Available` flags are an output from the implemented sensor blocks. Measurements that pass the first data validity check are then subjected to a χ^2 measurement association check in order to protect against corrupted measurements and filter smugness, a phenomenon which occurs when the estimated error covariance shrinks, causing the filter to effectively reject new measurements resulting in the diverging state estimate unless corrected [51]. Therefore, a given measurement is considered to be associated with the FSS to a user-defined probability when [49]

$$(\boldsymbol{z}_k - \boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k-1})^T(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1}(\boldsymbol{z}_k - \boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k-1}) \leq \zeta \tag{6}$$

where ζ is a χ^2-random variable with one degree of freedom, and $\boldsymbol{R}_k \in \mathbb{R}^{4\times4}$ is the sensor noise covariance matrix and is defined as follows:

$$\boldsymbol{R}_k = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 \\ 0 & 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & 0 & \sigma_{\dot{\theta}}^2 \end{bmatrix} \tag{7}$$

Note that, because each measurement is assumed to be independent, there is only one degree of freedom. Additionally, the default value chosen is 99.95%, which results in $\zeta_{99.95\%} = 12.116$. Although Eq. (6) is written in matrical form for compactness, it can be easily expanded into four scalar equations for easier comparison with the association threshold. Additionally, the measurement matrix does not contain any offdiagonal terms; the computations for each measurement are independent, therefore eliminating costly matrix inverses. If any measurement violates Eq. (6), that specific row in the measurement mapping matrix is set to all zeros and the respective bit in the DKF health telemetry item for gating is set to "1". Additionally, in an effort to overcome filter smugness, the diagonal term in the error covariance estimate is artificially inflated by a user-defined scalar for which the default value is 1.025, which is in a similar fashion to a fading-memory Kalman filter [50,52].

Once the measurement mapping matrix is created, the state estimate correction and error covariance correction are computed in the traditional manner where the Kalman gain, state estimate, and error covariance estimate are

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1} \tag{8}$$

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k-1}) \tag{9}$$

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I}_{6\times6} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k|k-1}(\boldsymbol{I}_{6\times6} - \boldsymbol{K}_k\boldsymbol{H}_k)^T + \boldsymbol{K}_k\boldsymbol{R}_k\boldsymbol{K}_k^T \tag{10}$$

Lastly, the FSS DKF is considered to be converged when the absolute value of the change in the square of the Frobenius norm of the measured diagonal elements of the error covariance estimate is below some user-defined threshold ϵ_{DKF}. This condition is shown in Eq. (11), where

$$\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{4} \boldsymbol{A}_{ii}^2}$$

denotes the Frobenius norm of matrix $\boldsymbol{A}$:

$$\|\boldsymbol{H}_k\boldsymbol{P}_{k|k}\boldsymbol{H}_k^T\|_F^2 - \|\boldsymbol{H}_{k-1}\boldsymbol{P}_{k-1|k-1}\boldsymbol{H}_{k-1}^T\|_F^2 \leq \epsilon_{\text{DKF}}^2 \tag{11}$$

Although the filter convergence condition in Eq. (11) is met, the associated bit field in the DKF health telemetry is set to "converged." The filter convergence status is used in two places in the FSS GNC. First, on test startup, the FSS air bearings and guidance subsystem are enabled only after the DKF filter is converged. Second, the state and error covariance estimates are only passed to the outer loop for use on the next DKF cycle when the filter is converged.

3. *Thruster Mapping and Modulation*

The control subsystem of every FSS includes the thruster mapping and thruster modulation algorithm. The purpose of thruster mapping is to select the appropriate thruster to fire in order to realize the desired control input consisting of forces and torques. Additionally, because the thrusters can only take on two discrete states (on and off) while the requested control inputs from the guidance and control algorithms are continuous, a modulation scheme is required.

The eight FSS thrusters are assumed to be identical for the sake of thruster mapping. The requested control input can be mapped to the appropriate thruster from:

$$\boldsymbol{u} = \boldsymbol{M}\boldsymbol{f} \tag{12}$$

where $\boldsymbol{u} = [F_x, F_y, \tau]^T$ denotes the control input, $\boldsymbol{f} = [f_1, f_2, ...f_8]^T$ denotes the force applied by each thruster, and $M \in \mathbb{R}^{3\times8}$ is the thruster to control input mapping matrix. Therefore, the force required by each thruster to realize the control input is

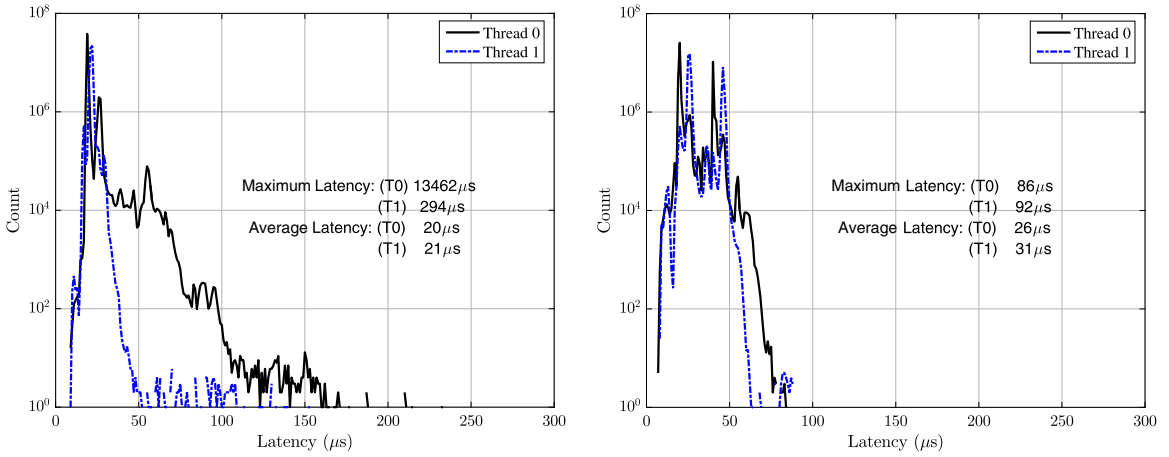
$$\boldsymbol{f} = 2\boldsymbol{M}^+\boldsymbol{u} \tag{13}$$

where $\boldsymbol{M}^+$ denotes the Moore–Penrose pseudoinverse of $\boldsymbol{M}$. Because the mapping matrix $\boldsymbol{M}$ includes both positive and negative assignments, the pseudoinverse results in equal scaling along both the positive and negative directions. As a result, the factor of two in Eq. (13) compensates for this fact in order to scale the requested force equally across only the applicable thrusters. For instance, if a force is requested along the $+x$ direction, thrusters 2 and 3 would be selected, with each applying half of the requested force.

After the requested control input has been appropriately assigned to specific thrusters, a modulation scheme is required to realize the force given the discrete nature of the thrusters. Several common pulse modulators used to modulate spacecraft thrusters are Schmitt triggers, pulse-width pulse-frequency modulators, derived-rate modulators, and pulse-width modulators (PWMs) [53]. Another method, which was initially considered by Ciarcia et al. and further

Table 3 Summary of the cyclic test latency results

Kernel type		Minimum latency, μs	Average latency, μs	Maximum latency, μs
Stock	Thread 0	9	20	13,462
	Thread 1	9	21	294
Patched	Thread 0	7	26	86
	Thread 1	8	31	92



a) Stock, unpatched Linux kernel **b) RT-Preempt patched Linux kernel**

Fig. 6 Comparison of operating system latencies between unpatched and patched Linux kernels.

investigated by Zappulla et al.,[**] is the use of a sigma-delta modulator ($\Sigma\Delta M$) for spacecraft thruster modulation [30]. The $\Sigma\Delta M$ was experimentally compared against the PWM and found to achieve more than two times less steady-state attitude error (3σ) and use only slightly more thruster on time than the PWM [30]. It is worth noting that several control subsystems with PWMs, $\Sigma\Delta M$s, and a $\Sigma\Delta$–PWM hybrid are implemented and available via a custom Simulink library for developers to select. The default modulator in the Simulink template, however, is set to the $\Sigma\Delta M$.

4. *Reaction Wheel Controller*

The reaction wheel is controlled though a simple speed-mode controller. In such a controller, the requested torque is converted, using the RW inertia, to a required flywheel acceleration and integrated to obtain an ideal flywheel velocity. The measured flywheel velocity ω_{Measured} is compared with the ideal velocity $\Delta\omega = \omega_{\text{Ideal}} - \omega_{\text{Measured}}$. The difference between flywheel velocities $\Delta\omega$ is scaled by a proportional gain in order to generate an analog voltage that drives the RW motor torque. The flywheel velocity is read, and the voltage is updated at a 20 Hz rate. An overview of this RW speed-mode controller is provided in Fig. 5.

This type of controller can compensate for the RW friction, and it controls the RW so that the correct amount of angular momentum is transferred between the RW and the FSS. To avoid the zero-crossing problem, the RW is spun up to about 1100 RPM (half of its maximum rated velocity) before the experiment starts.

C. Numerical Development Simulator

The purpose of the numerical development simulator is to create an environment that accurately represents the testbed, allowing for rapid development of guidance algorithms as a "virtual" testbed. Except for the interfaces from the sensors and to the control actuators, the numerical simulator uses the same supporting code (such as navigation filters, actuator control logic, and rate transitions) in order to achieve this goal. This layout simplifies the transition from the development environment to running the software onboard the FSS.

To create a representative interface to the DKF, both the Vicon and FOG sensors are modeled using an additive white noise model. The simulated Vicon sensor takes in the simulated truth position and attitude of the FSS, adds a zero mean Gaussian noise with a variance of the respective state found via sensor noise characterization, and outputs a noisy state at a user-defined rate. Likewise, the simulated FOG sensor takes in the truth angular rate of the FSS, adds several noise terms, and outputs the noisy state. The zero mean Gaussian noise sources include a rate noise density and an in-run bias stability. The in-run bias stability is integrated, given an initial bias, before being added with the rate noise to the true angular rate of the FSS to create the simulated FOG sensor output. The values for the rate noise density and in-run bias stability were obtained from the FOG sensor noise characterization (see Sec. III) and the KVH DSP-3000 FOG datasheet for values not found in the characterization.

[**]Zappulla II, R., Virgili-Llop, J., and Romano, M., "Sigma-Delta Modulation for Spacecraft Thruster Control," *IEEE Transactions on Control Systems Technology*, pp. 1–6 (under review).

III. Characterization

This section will detail the system identification a fourth-generation FSS. This characterization includes OS latency, sensor noise, thruster performance, mass, moment of inertia, and testbed residual acceleration.

A. OS Latency

By applying the RT-Preempt patch to the Linux kernel, the maximum latency the OS exhibits in response to a stimuli is bounded. To quantify the latency characteristics of both the stock (unpatched) Linux kernel and the RT-Preempt patched Linux kernel, the cyclic test was run for 50 million iterations [54]. The results are summarized in Table 3 and illustrated in Fig. 6. Compared to the stock Linux kernel, for which the maximum observed latency during the `Cyclictest` was over 13 ms, the RT-Preempt patched kernel bounded the OS latencies to within 100 μs. Furthermore, the maximum average latency for both the stock and patched kernels was observed to occur on thread 1. It is worthwhile to note that, although the average latency of the patched kernel was slightly larger in value than the stock kernel, it was of the same order of magnitude and, more importantly, the maximum latency was bounded to within 100 μs.

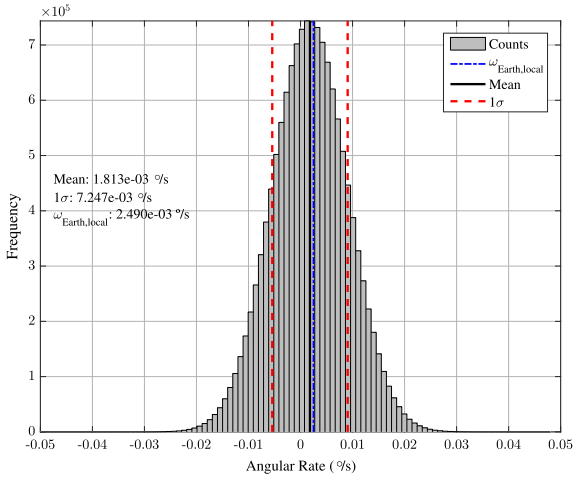
B. Sensor Noise Characterization

1. *Vicon Noise Characterization*

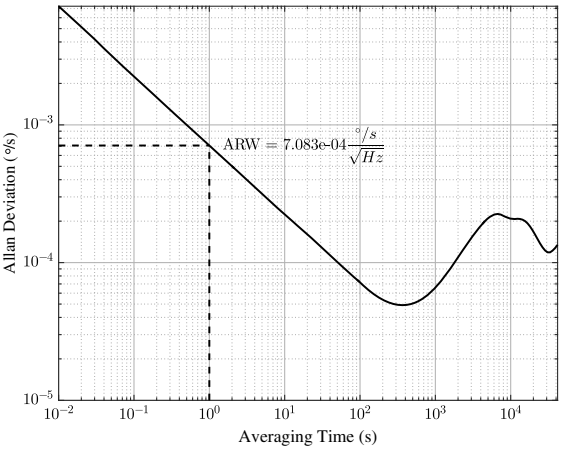
It is assumed that the noise associated with the measurements provided by the Vicon (x, y, and θ) are independent because only processed measurements are provided. This assumption enables the characterization of each measurement produced by the Vicon to be performed independent of the other measurements. To characterize the translational sensor noise, the FSS is given only a translational velocity across the diagonal of the granite monolith. Its position from the Vicon sensor is then recorded directly, and the resulting

Table 4 Vicon sensor noise characterization results

Trial	σ_x, m	σ_y, m	σ_θ, rad
1	0.01126	0.008889	0.03556
2	0.01179	0.01296	0.002549
3	0.001986	0.006620	0.02685
Mean	0.006412	0.009136	0.01345



a) FOG data histogram



b) FOG Allan variance plot

Fig. 7 FOG data histogram with mean output and 1σ indications, as well as Allan variance plot.

time histories for x and y are independently curve fit to both a first-order model $[x(t) = a_1 t + a_0]$ and a second-order model $[x(t) = a_2 t^2 + a_1 t + a_0]$, where $\{a_0,\ a_1,\ a_2\}$ are the model coefficients. Assuming the acceleration coefficient in the second-order model a_2 is sufficiently small, the root-mean-square error (RMSE) of the first-order error is interpreted as the noise parameter for the Vicon x and y measurements, respectively. Furthermore, in order to capture any effects of speed on sensor noise, the FSS was given three different speeds across the diagonal of the granite monolith. Lastly, the geometric mean of the three speeds was used to compute the resulting Vicon noise parameter for the x and y measurements. A similar method was followed to characterize the attitude measurement θ except, instead of a translational-only velocity, the FSS was given only a rotational velocity.

The resulting Vicon measurement noise values across each trial and the resulting geometric means are tabulated in Table 4. As illustrated, the Vicon sensor noise for both translational measurements are relatively close together, whereas the sensor noise for the attitude measurement is larger. This is expected because the attitude measurements are derived from the position of each of the markers located atop the FSSs, which form the basis of the Vicon measurement frame set in the Vicon software.

2. *FOG Noise Characterization*

To estimate the FOG noise parameters, an Allan variance analysis was performed following IEEE Standard 952-1997 [55]. The Allan variance method was initially used to investigate the frequency stability of precision oscillators in the time domain. However, this method has been extended to aid in characterizing random noise processes in a wide variety of devices [55,56].

To perform the static characterization of the FOG, first, over 13.75 million samples were collected at 100 Hz over 27 h from the FOG, as illustrated in Fig. 7a. A basic analysis of the static FOG data yielded the mean output, the standard deviation, as well as the quantization level. Although not relevant to the FOG noise characterization, the quantization level is useful for creating the simulated FOG data. It can be seen that the FOG is able to detect the rotation of the Earth, for which the value at the location of the testbed is 0.002490 deg /s.

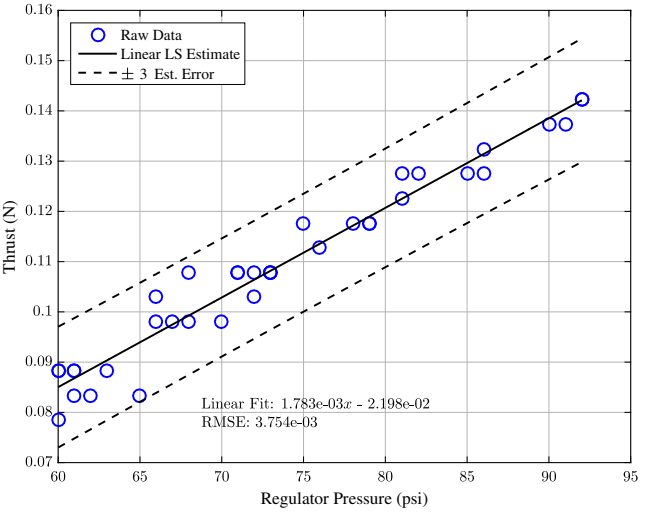


Fig. 8 FSS thruster characterization data with first-order fit and estimated (Est.) 3σ bounds.

Next, performing the Allan variance analysis yields the Allan variance plot for the FOG illustrated in Fig. 7b. The relevant noise parameter estimated from the Allan variance analysis is the angular random walk (ARW). The ARW is identified on the Allan variance plot as having a slope of $-1/2$. Furthermore, the value for the ARW is obtained by fitting a line through this segment on the plot and reading the value at $\tau = 1$ [55]. Given that the FOG operates at 100 Hz, the equivalent white noise value using the Allan variance method was found to be 7.083×10^{-3} deg /s. It is worthwhile to note that the white noise value derived from the Allan variance analysis is within 2.3% of the value derived from a statistical analysis of the FOG data. Another useful noise parameter that can be estimated via the Allan variance analysis is the bias stability. This parameter would appear on the plot as a segment with a zero slope. However, due to the large averaging times required, estimating this

Table 5 Summary of FOG analysis

Parameter	Value
Mean output	1.813×10^{-3} deg /s
White noise (statistical analysis)	7.247×10^{-3} deg /s
White noise (Allan variance)	7.083×10^{-3} deg /s
Quantization level	1×10^{-6} deg /s

Table 6 Summary of relevant FSS physical properties

Parameter	Value
Mass, wet	9.882 kg ± 0.001 kg
Mass, dry	9.465 kg ± 0.001 kg
Dimensions	$0.27 \times 0.27 \times 0.52$ m
Estimated MOI	0.2527 kg · m^2
MOI error (1σ)	0.0115 kg · m^2

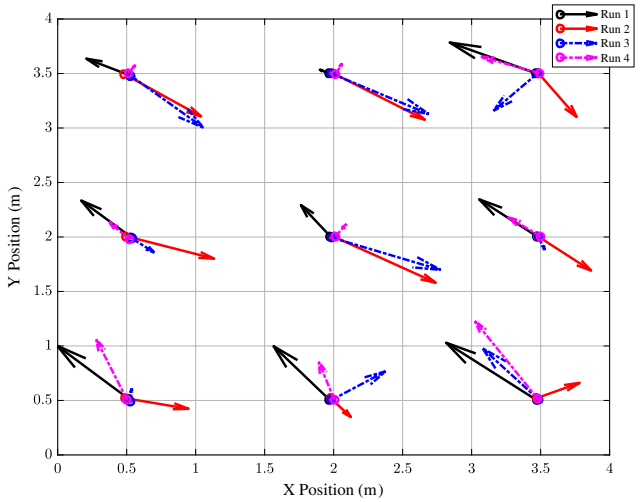


Fig. 9 Average residual acceleration at nine locations on the granite monolith.

parameter via this method is infeasible. Because this value is typically smaller than the ARW, and given the short duration of the tests, this parameter can be neglected.

A summary of relevant FOG parameters from the two analyses are listed in Table 5. The white noise value derived from the statistical analysis of the static FOG data to use in the DKF was determined to be 7.247×10^{-3} deg/s or, equivalently, 1.265×10^{-4} rad/s.

C. Thruster Characterization

Characterizing the performance of the thrusters is essential across several aspects of the FSS testbed. This includes accurate modeling in the numerical simulator, control input in the DKF, guidance parameter tuning, and moment of inertia (MOI) estimation. A single thruster was characterized in [36]; here, we present a characterization of the thruster performance when integrated into the FSS. Given the architecture of the system, the force generated by the thruster is highly influenced by the pressure regulator setting and the number of values open at the same time. In an effort to characterize the nominal performance of the thrusters, two thrusters on the same face are fired for several seconds and the maximum force generated by the thrusters is measured by a scale with a resolution of 9.81 mN. Additionally, it is assumed that the measured force has equal contributions from the two thrusters. Before firing the thrusters, the value reported by the pressure regulator, with a 1 psi resolution, is recorded such that a pressure versus thruster force mapping can be made. This process is repeated across a range of operating pressures and several times to ensure repeatability. The results are illustrated in Fig. 8. As illustrated, several regulator pressures took on similar thrust values due to the resolution of both the scale and pressure regulator.

Because no other data, such as temperature, are available, a linear fit is made between the regulator pressure and thrust generated. This linear model can then be used in the MOI estimation, as well as adjusting the thrust parameter for the DKF and simulator. It is important to note that the GNC software does not receive pressure information from the regulator, which precludes the use of the linear model onboard for estimating thrust.

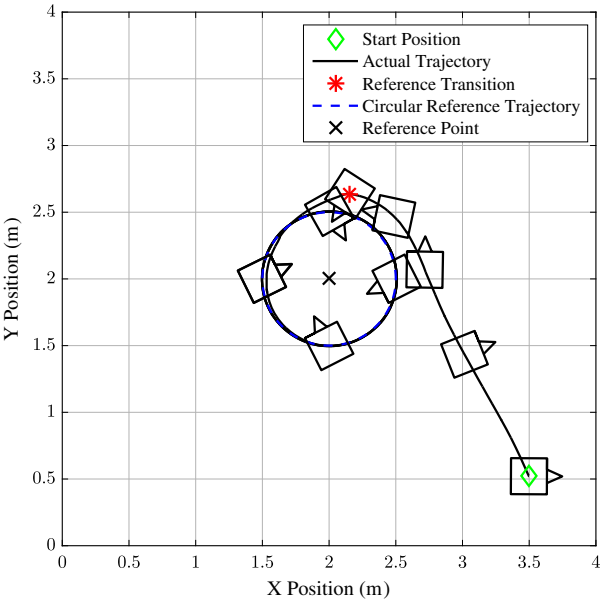


Fig. 10 Overhead view of the FSS path with four equally spaced attitude snapshots in time across the tangential intercept guidance segment and a single revolution about the fixed point.

D. Mass and Moment of Inertia

Estimating the physical properties (mass and MOI) is critical to not only understanding the system dynamics but also to creating the numerical simulator and developing and tuning the guidance algorithms. The system masses of a FSS are tabulated in Table 6.

To estimate the FSS MOI, a torque was first applied to the stationary vehicle in the $-\theta$ direction by the thrusters, followed by a coast period; lastly, an opposite torque was applied in the $+\theta$ direction by the thrusters to spin the vehicle down. It is worthwhile to note that the value indicated by the thruster pressure regulator was recorded before the test in order to use the linear model to estimate both the thrust output and torque of each thruster. The resulting attitude time history was recorded directly from the Vicon sensor. Next, the portions of the attitude time history under thrusting were fit to the second-order model $\theta(t) = a_2 t^2 + a_1 t + a_0$, where $a_2 = 1/2(\tau/J_{zz})$, from which the MOI (J_{zz}) is estimated given the estimated torque τ. This test was performed five times in order to gather 10 attitude time histories under thrust, refueling the FSS each time prior to running the test. The resulting MOI estimate is tabulated in Table 6.

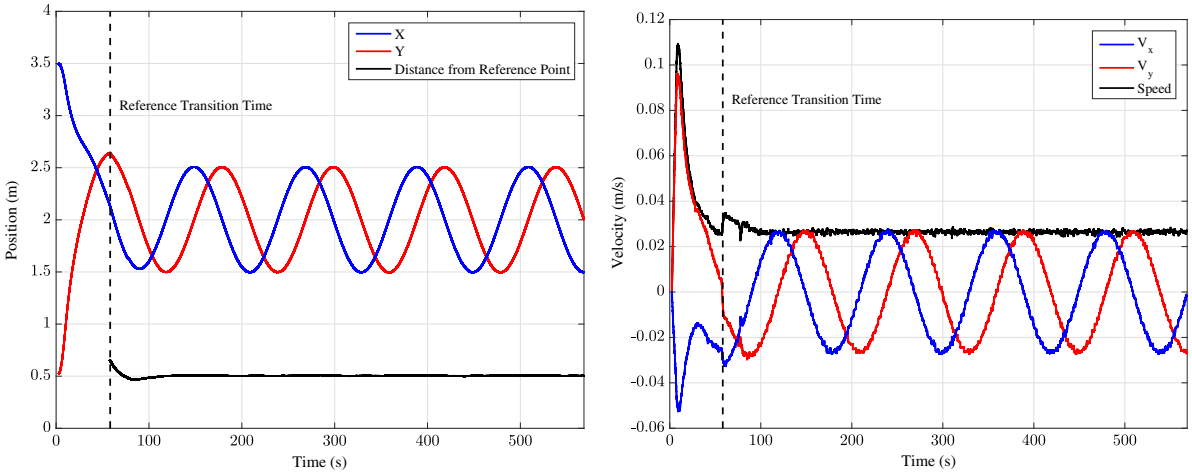
E. Residual Acceleration

From an experimental standpoint, it is important to develop a bounded estimate of the residual accelerations acting on the FSS during a test campaign. In addition to any component of gravity in the plane of motion and residual friction, these unmodeled accelerations are also due to lateral air-bearing leakage due to the air bearing not exactly being parallel to the testbed surface and the

Table 7 Case study test parameters

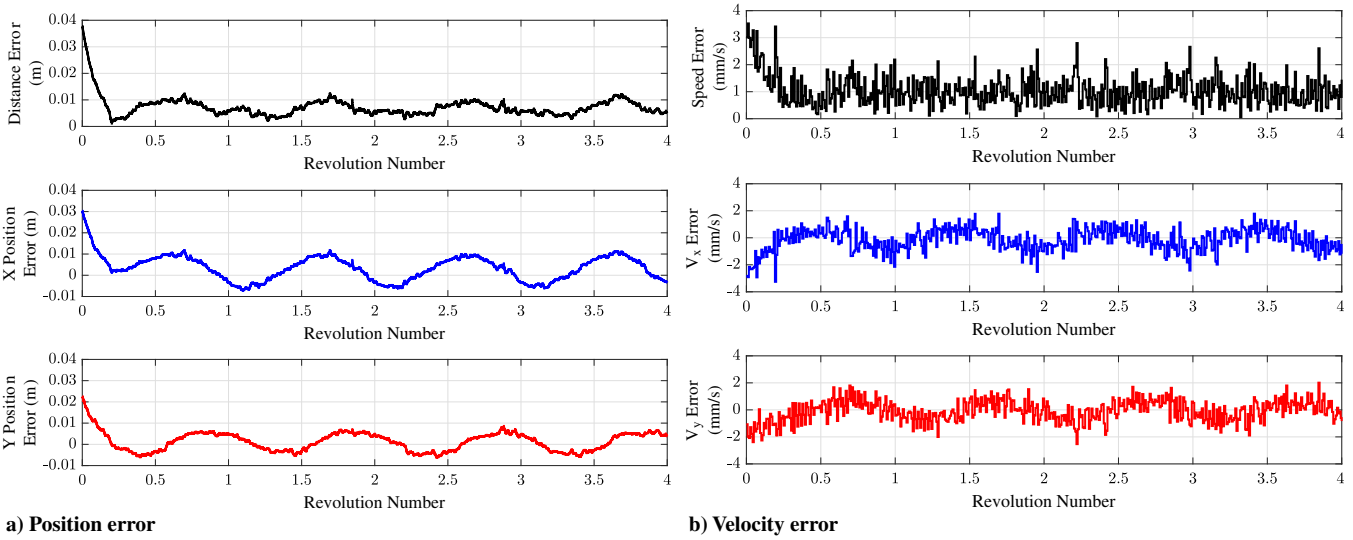
Guidance parameters	Values	Navigation parameters	Values	Control parameters	Values
Sample time	0.10 s	Sample time	0.02 s	Sample time	0.01 s
(x_c, y_c)	(2, 2) m	q_k^2	0.001	$\Sigma\Delta M$ sample time	0.10 s
r_{ref}	0.5 m	σ_x	0.0064123 m	Thruster force	0.115 N
$\dot{\psi}_{\text{ref}}$	3 deg/s	σ_y	0.0091363 m	Minimum firing time	0.06 s
Number of revolutions	4	σ_θ	0.013451 rad	Minimum impulse bit	0.0115 N · s
Q_a	diag(0.05,0.05,0.05)	$\sigma_{\dot{\theta}}$	0.00012649 rad/s	— —	— —
K_a	diag(1,1,0.1)	ζ	12.116	— —	— —
— —	— —	ϵ^2_{DKF}	5×10^{-8}	— —	— —

airflow coming out of the air bearing being not exactly symmetric [13]. In an attempt to quantify these residual accelerations, a FSS was positioned at the nine locations illustrated in Fig. 9 on the granite monolith via an onboard Linear Quadratic Regulator (LQR) controller. Upon arriving at a predefined grid point, the air pads were closed for 30 s, allowing the FSS and the DKF to stabilize

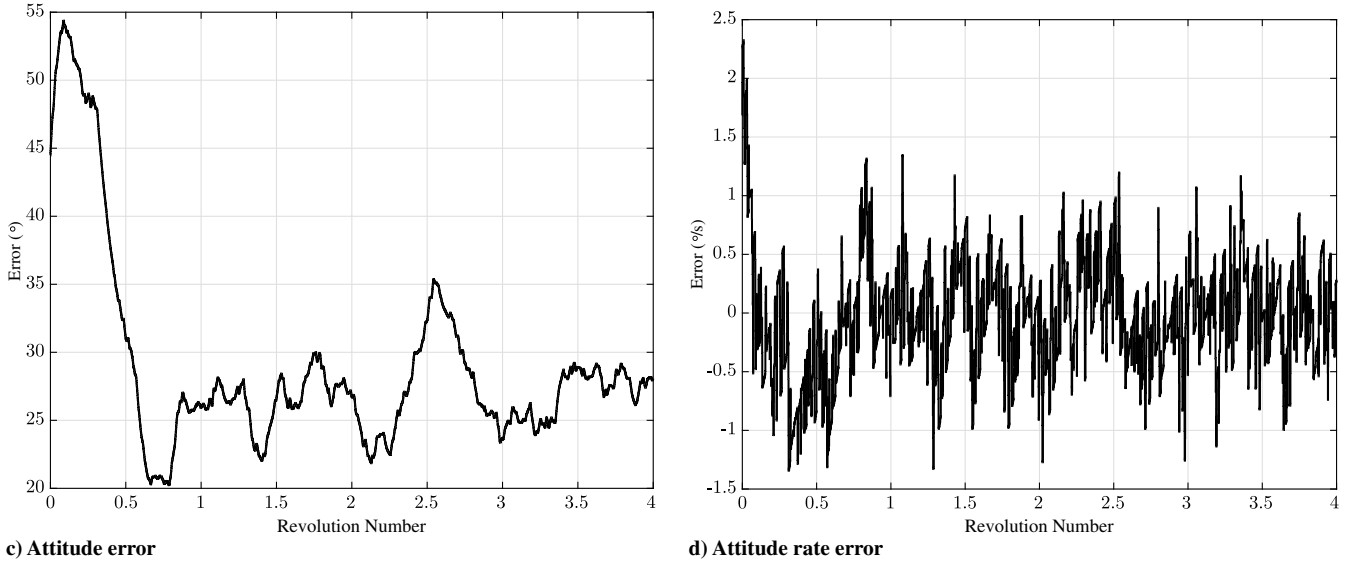


a) Position **b) Velocity**

Fig. 11 Position and velocity of the FSS during the entire maneuver.



a) Position error **b) Velocity error**



c) Attitude error **d) Attitude rate error**

Fig. 12 Position and velocity errors of the FSS during circumnavigation.

Table 8 Summary of the circumnavigation time, position, speed, and attitude errors

	Revolution				
	1	2	3	4	Overall
Revolution time, s	120.00	120.00	120.00	119.91	119.98
Mean position error, mm	9.18	6.56	6.64	6.64	7.25
Mean speed error, mm/s	1.18	1.01	1.01	0.99	1.04
Mean attitude error, deg	35.07	26.64	27.62	27.01	29.08
Mean attitude rate error, deg/s	0.52	0.34	0.35	0.30	0.38

before opening again for 30 s to perform the test at the location. The resulting acceleration imposed on the FSS during the 30 s interval was then derived from onboard state estimate information. The average (scaled) accelerations at each location over four tests is illustrated in Fig. 9. The overall average translational and rotational residual accelerations over the nine locations was found to be 1.871×10^{-4} m/s^2 (19.1 μG) and 7.56×10^{-2} deg/s^2, respectively.

IV. Circumnavigation Case Study

A. Overview

To demonstrate the capabilities of the POSEIDYN testbed, an exemplary case study is performed where the FSS enters into and maintains a circular path with a 0.50 m radius and a tangential velocity for four revolutions, at a constant angular velocity of 3 deg/s about the center, using an artificial potential function (APF)-based guidance law to control the FSS [28,57,58].

B. Guidance and Control Development

Assuming the FSS will start outside of the desired circular reference path at some point and attitude $\boldsymbol{x} = [x, y, \theta]^T$, the guidance will initially target a point that is tangential to the current position of the FSS and the circular reference trajectory:

$$\boldsymbol{p}_f = r_{\text{ref}} \begin{bmatrix} -\sin(\phi - \beta) \\ \cos(\phi - \beta) \end{bmatrix} + \boldsymbol{p}_{\text{ref}} \tag{14}$$

where $\boldsymbol{p}_{\text{ref}} = [x_c, y_c]^T$ is the center of the reference trajectory, r_{ref} is the radius of the circular reference trajectory, $\beta = \sin^{-1}(r_{\text{ref}}/D)$ is the angle between the FSS and a point tangent to the reference trajectory, D is the distance from the FSS to the center of the reference trajectory, and $\phi = \tan^{-1}((y_c - y)/(x_c - x))$ is the angle of the FSS relative to the center of the reference trajectory in the inertial frame. During this segment, only the reference position is passed to the APF-based control law. Furthermore, the FSS is commanded to point the $+x$ face toward the tangent intercept point. It is worthwhile to note that attitude control is performed via thruster actuation.

Once the FSS is within a user-defined distance to the circular reference trajectory, the guidance targets the circular reference trajectory and sets the initial angle ψ_0 of the FSS relative to the center of

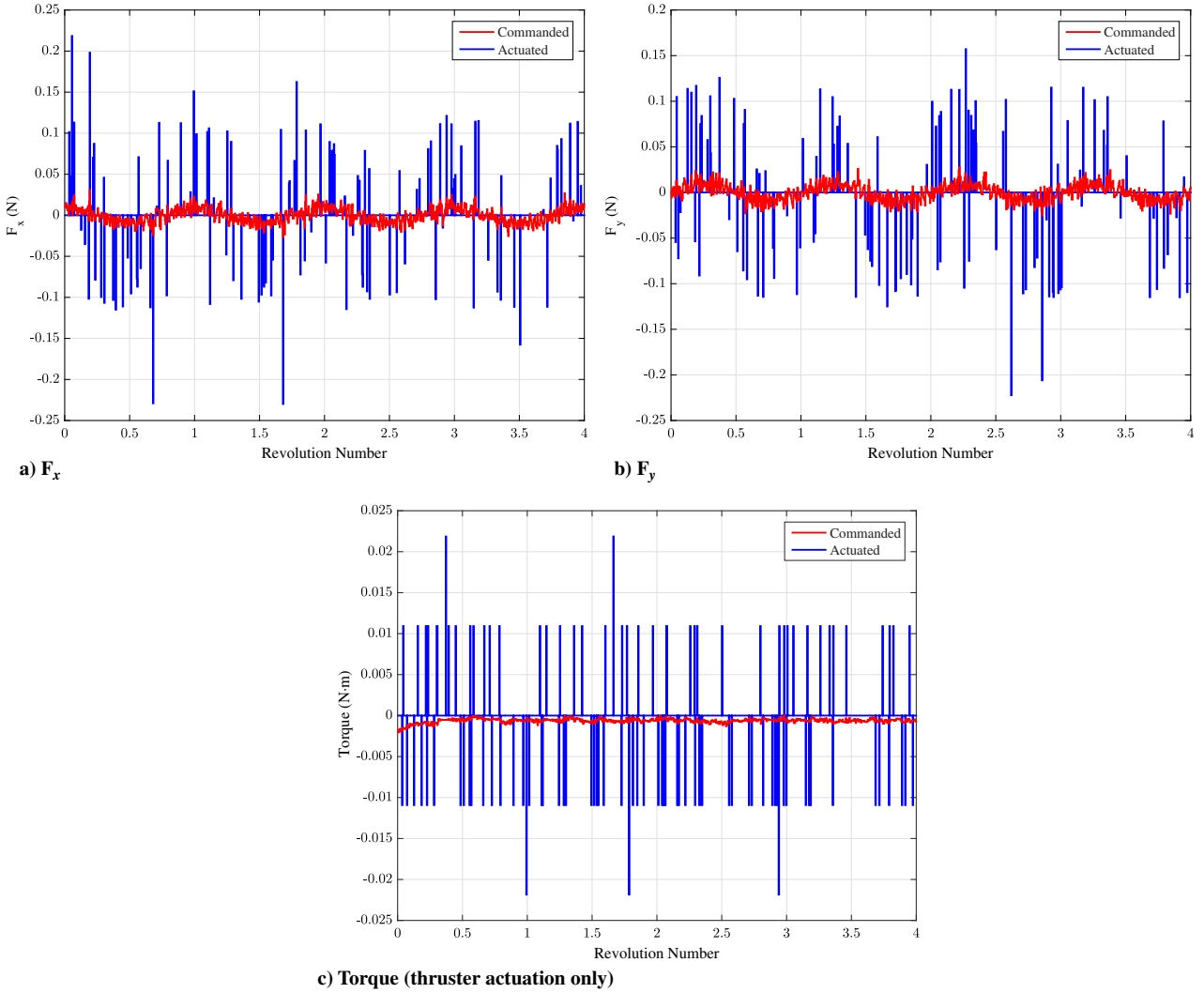
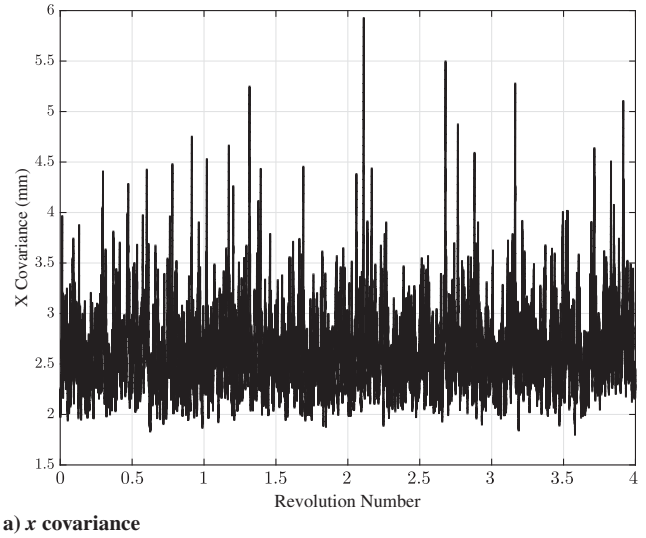


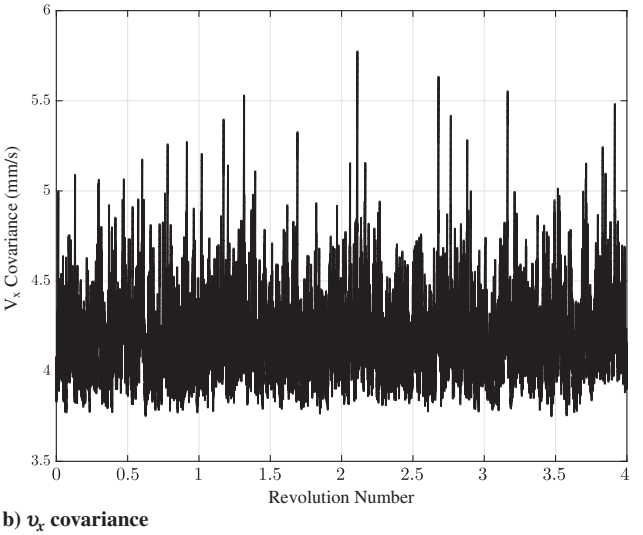
Fig. 13 Commanded and actuated control inputs (in the inertial frame) throughout the circumnavigation maneuver.

the reference trajectory in the inertial x direction at this point in time. Subsequent reference positions and velocities are determined by propagating the initial reference angle at the user-specified rate $\dot{\psi}_{\text{ref}}$:

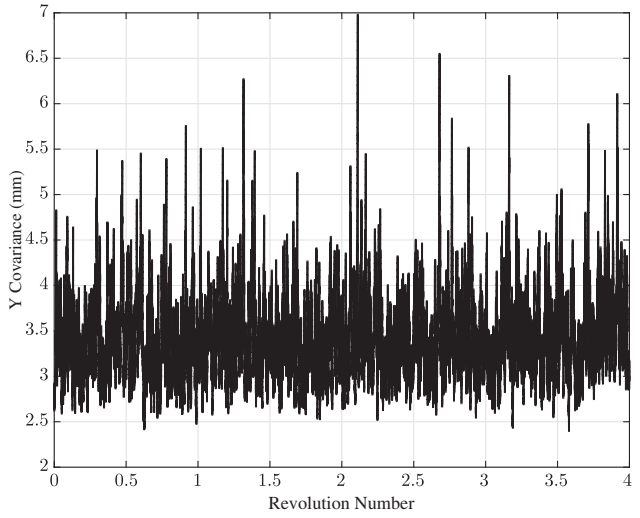
$$\boldsymbol{p}_f = \boldsymbol{p}_{\text{ref}} + r_{\text{ref}} \begin{bmatrix} \cos(\psi) \\ \sin(\psi) \end{bmatrix} \tag{15}$$



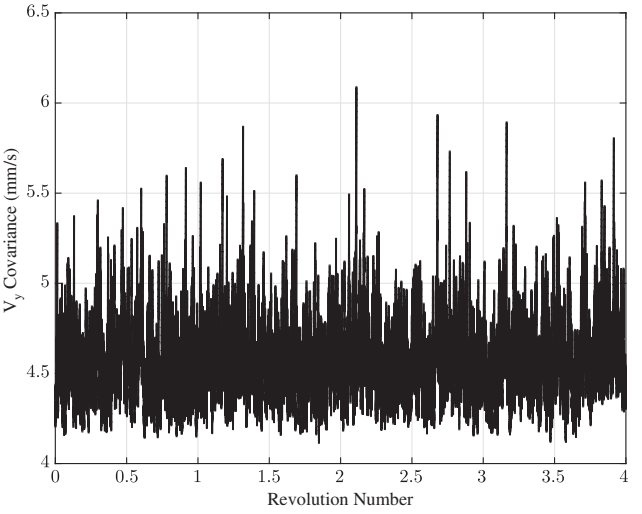
a) x covariance



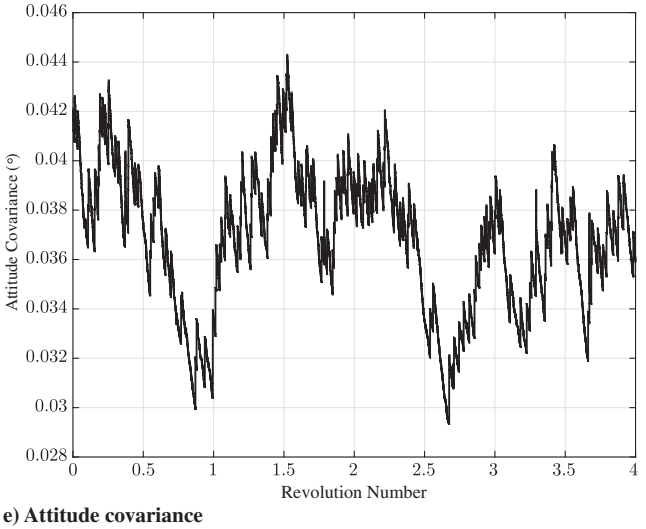
b) υ_x covariance



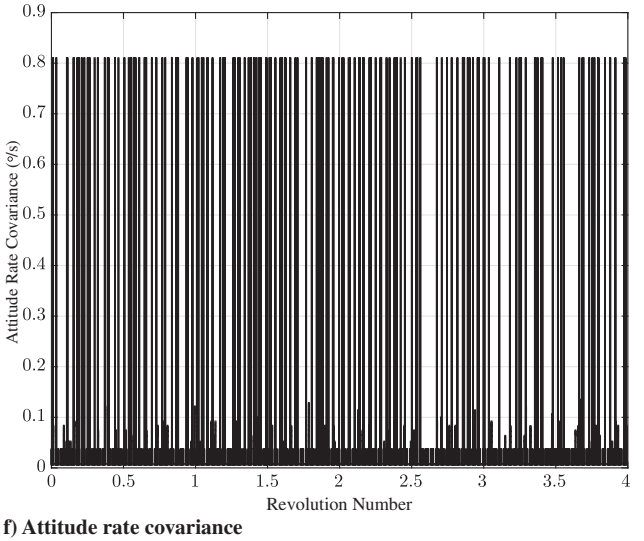
c) y covariance



d) υ_y covariance



e) Attitude covariance



f) Attitude rate covariance

Fig. 14 Covariance time histories throughout the circumnavigation maneuver.

Table 9 Mean estimated error standard deviation over the circumnavigation maneuver

	Revolution				
	1	2	3	4	Overall
x, mm	2.26	2.63	2.64	2.65	2.63
v_x, mm/s	4.19	4.20	4.21	4.21	4.21
y, mm	3.39	3.41	3.43	3.44	3.42
v_y, mm/s	4.58	4.59	4.60	4.60	4.59
θ, deg	0.037	0.038	0.036	0.036	0.037
ω, deg /s	0.016	0.016	0.015	0.014	0.015

$$\dot{\boldsymbol{p}}_f = r_{\text{ref}}\dot{\psi}_{\text{ref}}\begin{bmatrix} -\sin(\psi) \\ \cos(\psi) \end{bmatrix} \tag{16}$$

During the circumnavigation portion of the maneuver, the $+x$ face of the FSS is pointed inward toward the fixed reference point,

$$\theta_{\text{ref}} = \tan^{-1}\left(\frac{y_c - y}{x_c - x}\right) \tag{17}$$

An APF control law is used to generate the desired commands to follow the reference trajectory and reference attitude. To generate the commands, a potential field is setup using only one quadratic attractive potential located at the reference position and attitude

$$\Phi_a = \frac{1}{2}\boldsymbol{K}_a(\boldsymbol{x}-\boldsymbol{x}_{\text{ref}})^T\boldsymbol{Q}_a(\boldsymbol{x}-\boldsymbol{x}_{\text{ref}}) \tag{18}$$

where $K_a \in \mathbb{R}^{3\times3}$ is a small, symmetric, positive gain matrix; and $Q_a \in \mathbb{R}^{3\times3}$ is the symmetric, attractive potential shaping matrix. It is worthwhile to note that, if additional obstacles or keepout zones were present, repulsive potentials (i.e., a source) could be placed at these locations to ensure obstacle avoidance and keepout zone enforcement. The resulting control law is given as the gradient of the attractive potential (and repulsive potential, if used):

$$\boldsymbol{u} = -\boldsymbol{B}^{-1}\boldsymbol{K}_a(\boldsymbol{Q}_a(\boldsymbol{x}-\boldsymbol{x}_{\text{ref}}) + (\dot{\boldsymbol{x}}-\dot{\boldsymbol{x}}_{\text{ref}})) \tag{19}$$

where $B \in \mathbb{R}^{3\times3}$ is the lower half of the continuous-time input gain matrix.

C. Case Study Setup

A list of the relevant test parameters is reported in Table 7. It is worthwhile to note that, before running the test on the POSEIDYN testbed, the typical development workflow included preliminary testing using the development numerical simulator to tune and debug the guidance algorithm.

D. Case Study Results and Discussion

As illustrated in Figs. 10 and 11, the FSS starts in the lower right-hand corner of the granite monolith. At the start of the test, the guidance subsystem targets a point tangent to the circular reference trajectory. It can be seen during the tangent point targeting segment that the FSS attitude only achieves a near-zero attitude error as it nears the reference trajectory transition (marked by an asterisk "$*$" in Fig. 10) due to a small attractive potential weighting on the attitude error. Once the FSS gets sufficiently close to the circular reference trajectory, the reference trajectory switches and the FSS is seen to track the desired circular trajectory fairly well. As illustrated in Fig. 12, the maximum distance from the circular reference trajectory was less than approximately 0.04 m. The maximum difference between the FSS speed and the reference speed was no larger than approximately 0.003 m/s, except for a brief period of time when the FSS was establishing on the reference trajectory. Furthermore, a relatively constant attitude error throughout the circumnavigation can be seen in Figs. 10 and 12c. This is attributed to the small attractive potential weighting on the attitude error used in the case study. The time, position, speed, and attitude errors across each revolution and the entire maneuver are summarized in Table 8. Lastly, as anticipated, frequent actuated control inputs occur throughout the maneuver to maintain the circular reference trajectory, as illustrated in Fig. 13.

To gauge the performance of the FSS DKF, both the navigation health telemetry and the estimated error covariances must be examined. The estimated error covariances throughout the duration of the maneuver are illustrated in Fig. 14 and summarized in Table 9. As a result of the frequent thruster firings required to track the desired trajectory, as illustrated in Fig. 13, this case study tests the performance of the DKF. Specifically, the DKF is given minimal time to filter the noise introduced by the thruster firings. Additionally, the effects of increased process noise on the attitude rate covariance are illustrated in Fig. 15. Recall that, whenever a thruster is actuated, the navigation filter automatically inflates the process noise to account for uncertainties in the thruster model. Despite the process noise associated with the attitude rate state being inflated by a factor of 500,

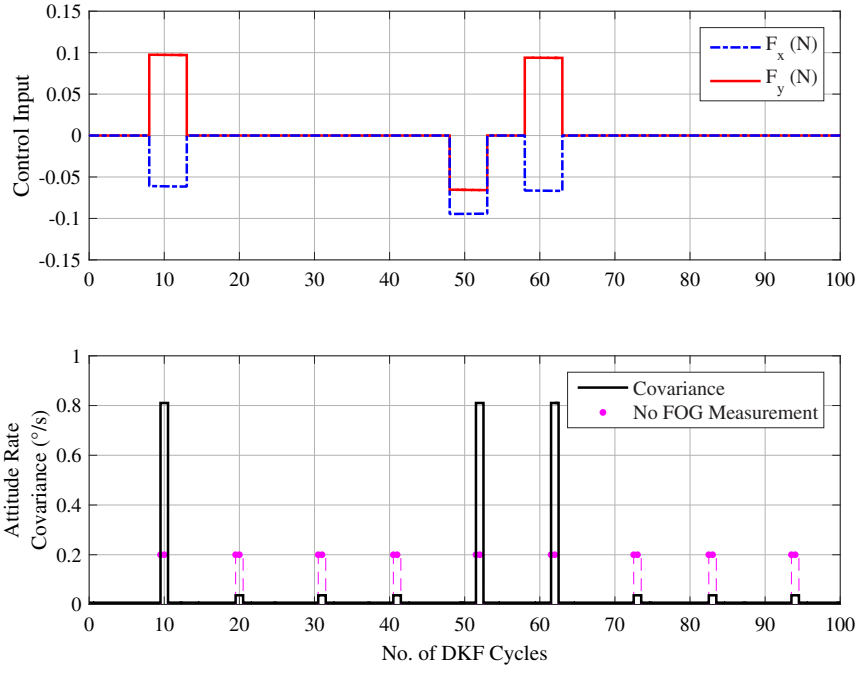


Fig. 15 Detailed illustration of the attitude rate covariance recovery.

the attitude rate covariance is able to recover quickly (within one to two DKF cycles) due to the small sensor noise associated with the FOG. Lastly, the small, periodic increases in the attitude rate covariance in between thruster firings were due to unavailable FOG sensor data, as indicated by the DKF health telemetry.

V. Conclusions

A detailed description of the Naval Postgraduate School's POSEIDYN testbed is provided. The quasi-frictionless and low residual acceleration dynamic behavior of the test vehicles operating on the POSEIDYN testbed are limited to two translational degrees of freedom and one rotational degree-of-freedom motion. Additionally, multiple vehicles can be operated simultaneously on the POSEIDYN testbed. This capability can be used to conduct research on coordinated control of spacecraft teams. The onboard actuators, composed of eight thrusters and/or a reaction wheel, are equivalent to the actuators found in orbital spacecraft and further enhance the dynamic equivalence. With an available simulator and extensive software development tools, guidance and control algorithms for real-time execution on board the test vehicles can be quickly and easily developed. Used as a last stage of on-the-ground validation before onorbit deployment, or simply as a more realistic development environment for novel guidance and control approaches, this state-of-the-art dynamic hardware-in-the-loop testbed will continue to be fruitful for the advancement of spacecraft proximity operations research.

References

[1] Quadrelli, M. B., Wood, L. J., Riedel, J. E., McHenry, M. C., Aung, M., Cangahuala, L. A., Volpe, R. A., Beauchamp, P. M., and Cutts, J. A., "Guidance, Navigation, and Control Technology Assessment for Future Planetary Science Missions," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 7, 2015, pp. 1165–1186. doi:10.2514/1.G000525

[2] "NASA Technology Roadmaps TA 4: Robotics and Autonomous Systems," NASA TR, 2015, https://www.nasa.gov/sites/default/files/atoms/files/2015_nasa_technology_roadmaps_ta_4_robotics_and_autonomous_systems_final.pdf.

[3] Davis, T. M., and Melanson, D., "XSS-10 Microsatellite Flight Demonstration Program Results," *Defense and Security*, Vol. 5419, International Society for Optics and Photonics, Bellingham, WA, April 2004, pp. 16–25.

[4] Zimpfer, D., Kachmar, P., and Tuohy, S., "Autonomous Rendezvous, Capture and In-Space Assembly: Past, Present and Future," *1st Space Exploration Conference: Continuing the Voyage of Discovery, Space Exploration Conferences*, Vol. 1, Orlando, FL, 2005, pp. 234–245. doi:10.2514/6.2005-2523

[5] Howard, R. T., and Bryan, T. C., "DART AVGS Flight Results," *Defense and Security Symposium*, International Society for Optics and Photonics, Bellingham, WA, 2007, Paper 65550L.

[6] Bosse, A. B., et al., "SUMO: Spacecraft for the Universal Modification of Orbits," *Defense and Security*, International Society for Optics and Photonics, Bellingham, WA, 2004, pp. 36–46.

[7] Henshaw, C. G., "The DARPA Phoenix Spacecraft Servicing Program: Overview and Plans for Risk Reduction," *International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2014)*, Montreal, 2014, http://robotics.estec.esa.int/i-SAIRAS/isairas2014/Data/Session%205b/ISAIRAS_FinalPaper_0043.pdf.

[8] Gunn, L., Palmer, J., Leung, J., Ogilvie, A., Tadros, A., Chappell, L., Weldy, K., Fowler, E., Eagen, J., and Sullivan, B., "DARPA Phoenix Payload Orbital Delivery System: Progress Towards Small Satellite Access to GEO," *29th Annual AIAA/USU Conference on Small Satellites*, SSC15-II-1, Logan,UT, July 2015.

[9] 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. doi:10.2514/1.40161

[10] Kjellberg, H. C., and Lightsey, E. G."Discretized Constrained Attitude Pathfinding and Control for Satellites," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 5, 2013, pp. 1301–1309. doi:10.2514/1.60189

[11] Schulte, P. Z., and Spencer, D. A., "Development of an Integrated Spacecraft Guidance, Navigation, and Control Subsystem for Automated Proximity Operations," *Acta Astronautica*, Vol. 118, Jan.–Feb. 2016, pp. 168–186. doi:10.1016/j.actaastro.2015.10.010

[12] Spencer, D. A., Chait, S. B., Schulte, P. Z., Okseniuk, K. J., and Veto, M., "Prox-1 University-Class Mission to Demonstrate Automated Proximity Operations," *Journal of Spacecraft and Rockets*, Vol. 53, No. 5, 2016, pp. 847–863.

[13] Rybus, T., and Seweryn, K., "Planar Air-Bearing Microgravity Simulators: Review of Applications, Existing Solutions and Design Parameters," *Acta Astronautica*, Vol. 120, March–April 2016, pp. 239–259. doi:10.1016/j.actaastro.2015.12.018

[14] Flores-Abad, A., Ma, O., Pham, K., and Ulrich, S., "A Review of Space Robotics Technologies for On-Orbit Servicing," *Progress in Aerospace Sciences*, Vol. 68, July 2016, pp. 1–26. doi:10.1016/j.paerosci.2014.03.002

[15] Gallardo, D., Bevilacqua, R., and Rasmussen, R. E., "Advances on a 6 Degrees of Freedom Testbed for Autonomous Satellites Operations," *AIAA Guidance, Dynamics and Control Conference*, AIAA Paper 2011-6591, 2011. doi:10.2514/6.2011-6591

[16] Saulnier, K., Pérez, D., Huang, R., Gallardo, D., Tilton, G., and Bevilacqua, R., "A Six-Degree-of-Freedom Hardware-in-the-Loop Simulator for Small Spacecraft," *Acta Astronautica*, Vol. 105, No. 2, 2014, pp. 444–462. doi:10.1016/j.actaastro.2014.10.027

[17] Wilde, M., Kaplinger, B., Go, T., Gutierrez, H., and Kirk, D., "ORION: A Simulation Environment for Spacecraft Formation Flight, Capture, and Orbital Robotics," *2016 IEEE Aerospace Conference*, 2016, IEEE, Piscataway, NJ, pp. 1–14. doi:10.1109/AERO.2016.7500575

[18] Tsiotras, P., "ASTROS: A 5DOF Experimental Facility for Research in Space Proximity Operations," *Proceedings of the Guidance, Navigation, and Control 2014*, AAS Paper AAS 14-114, Univelt Inc., San Diego, CA, 2014, pp. 717–730.

[19] Sabatini, M., Farnocchia, M., and Palmerini, G. B., "Design and Tests of a Frictionless 2D Platform for Studying Space Navigation and Control Subsystems," *2012 IEEE Aerospace Conference*, IEEE, Piscataway, NJ, 2012, pp. 1–12. doi:10.1109/AERO.2012.6187259

[20] Romano, M., Friedman, D. A., and Shay, T. J., "Laboratory Experimentation of Autonomous Spacecraft Approach and Docking to a Collaborative Target," *Journal of Spacecraft and Rockets*, Vol. 44, No. 1, 2007, pp. 164–173. doi:10.2514/1.22092

[21] Hall, J. S., and Romano, M., "Novel Robotic Spacecraft Simulator with Mini-Control Moment Gyroscopes and Rotating Thrusters," *2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, IEEE, Piscataway, NJ, 2007, pp. 1–6.

[22] Hall, J. S., and Romano, M., "Laboratory Experimentation of Guidance and Control of Spacecraft During On-Orbit Proximity Maneuvers," *Mechatronic Systems Simulation Modeling and Control*, edited by Paola, A. M. D. D., and Cicirelli, G., InTech, http://www.intechopen.com/books/mechatronic-systems-simulation-modeling-and-control/laboratory-experimentation-of-guidance-and-control-of-spacecraft-during-on-orbit-proximity-maneuvers.

[23] Virgili-Llop, J., Drew, J., and Romano, M., "Design and Parameter Identification by Laboratory Experiments of a Prototype Modular Robotic Arm for Orbiting Spacecraft Applications," *6th International Conference on Astrodynamics Tools and Techniques (ICATT)*, Darmstadt, Germany, March 2016, https://indico.esa.int/indico/event/111/session/34/contribution/75/material/paper/0.pdf.

[24] Curti, F., Romano, M., and Bevilacqua, R., "Lyapunov-Based Thrusters' Selection for Spacecraft Control: Analysis and Experimentation," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 4, 2010, pp. 1143–1160. doi:10.2514/1.47296

[25] Bevilacqua, R., Lehmann, T., and Romano, M., "Development and Experimentation of LQR/APF Guidance and Control for Autonomous Proximity Maneuvers of Multiple Spacecraft," *Acta Astronautica*, Vol. 68, No. 7, 2011, pp. 1260–1275. doi:10.1016/j.actaastro.2010.08.012

[26] Bevilacqua, R., Romano, M., Curti, F., Caprari, A. P., and Pellegrini, V., "Guidance Navigation and Control for Autonomous Multiple Spacecraft Assembly: Analysis and Experimentation," *International Journal of Aerospace Engineering*, Vol. 2011, 2011, Paper 308245 doi:10.1155/2011/308245

[27] Ciarcia, M., Grompone, A., and Romano, M., "A Near-Optimal Guidance for Cooperative Docking Maneuvers," *Acta Astronautica*, Vol. 102, Sept.–Oct. 2014, pp. 367–377. doi:10.1016/j.actaastro.2014.01.002

[28] Zappulla, R., Park, H., Virgili-Llop, J., and Romano, M., "Experiments on Autonomous Spacecraft Rendezvous and Docking Using an Adaptive Artificial Potential Field Approach," *Proceedings of the Space Flight Mechanics Meeting 2016*, AAS Paper AAS 16-459, Univelt Inc., San Diego, CA, Feb. 2016, pp. 4477–4494.

[29] Virgili-Llop, J., Zagaris, C., Park, H., Zappulla, R., and Romano, M., "Experimental Evaluation of Model Predictive Control and Inverse Dynamics Control for Spacecraft Proximity and Docking Maneuvers," *6th International Conference on Astrodynamics Tools and Techniques*, Darmstadt, Germany, March 2016, https://indico.esa.int/indico/event/111/session/38/contribution/32/material/paper/0.pdf.

[30] Ciarcia, M., Cristi, R., and Romano, M., "Experimental Emulation of the Scaled Clohessy-Wiltshire Dynamics on a Flat Air-Bearing Testbed," *AIAA Guidance, Navigation, and Control Conference, AIAA SciTech Forum*, AIAA 2017-1047, 2017. doi:10.2514/6.2017-1047

[31] Park, H., Zappulla, R., II., Zagaris, C., Virgili-Llop, J., and Romano, M., "Nonlinear Model Predictive Control for Spacecraft Rendezvous and Docking with a Rotating Target," *27th AAS/AIAA Spaceflight Mechanics Meeting*, AAS Paper 17-496, San Antonio, TX, Feb. 2017, http://hdl.handle.net/10945/51984.

[32] Zappulla, R., II., and Romano, M., "A Systematic Approach to Determining the Minimum Sampling Rate for Real-Time Spacecraft Control," *27th AAS/AIAA Spaceflight Mechanics Meeting*, AAS Paper 17-424, San Antonio, TX, Feb. 2017, http://hdl.handle.net/10945/51982.

[33] Virgili-Llop, J., Zagaris, C., Zappulla, R., II., Bradstreet, A., and Romano, M., "Convex Optimization for Proximity Maneuvering of a Spacecraft with a Robotic Manipulator," *27th AAS/AIAA Spaceflight Mechanics Meeting*, AAS Paper 17-418, San Antonio, TX, Feb. 2017, http://hdl.handle.net/10945/51983.

[34] Wilde, M., Ciarcià, M., Grompone, A., and Romano, M., "Experimental Characterization of Inverse Dynamics Guidance in Docking with a Rotating Target," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 6, 2016, pp. 1173–1187.

[35] Ciarcià, M., Cristi, R., and Romano, M., "Experimental Emulation of the Scaled Clohessy-Wiltshire Dynamics on a Flat Air-Bearing Testbed," *AIAA Guidance, Navigation, and Control Conference, AIAA SciTech Forum*, AIAA Paper 2017-1047, 2017. doi:10.2514/6.2017-1047

[36] Lugini, C., and Romano, M., "A Ballistic-Pendulum Test Stand to Characterize Small Cold-Gas Thruster Nozzles," *Acta Astronautica*, Vol. 64, No. 5, 2009, pp. 615–625. doi:10.1016/j.actaastro.2008.11.001

[37] Berger, R. W., Bayles, D., Brown, R., Doyle, S., Kazemzadeh, A., Knowles, K., Moser, D., Rodgers, J., Saari, B., Stanley, D., and Grant, B., "The RAD750-a Radiation Hardened PowerPC Processor for High Performance Spaceborne Applications," *2001 IEEE Aerospace Conference Proceedings (Cat. No. 01TH8542)*, Vol. 5, IEEE, Piscataway, NJ, 2001, pp. 2263–2272. doi:10.1109/AERO.2001.931184

[38] *Proton200k DSP Processor Board Datasheet*, Space Micro, Inc., San Diego, CA, 2014, http://www.spacemicro.com/assets/datasheets/digital/slices/proton200k-dsp.pdf.

[39] *Proton200k Lite Processor Board Datasheet*, Space Micro, Inc., San Diego, CA, 2015, http://www.spacemicro.com/assets/datasheets/digital/slices/proton200k-L.pdf.

[40] Kolmas, J., Banazadeh, P., Koenig, A. W., Macintosh, B., and D'Amico, S., "System Design of a Miniaturized Distributed Occulter/Telescope for Direct Imaging of Star Vicinity," *2016 IEEE Aerospace Conference*, Big Sky, MT, 2016, pp. 1–11. doi:10.1109/AERO.2016.7500783

[41] "NVIDIA Tegra K1 Series Processors with Kepler Mobile GPU for Embedded Applications Data Sheet," Ver. 04, NVIDIA, Santa Clara, CA, 2016.

[42] *Intel Core i7-4710MQ Processor*, Intel Corp., Santa Clara, CA, http://ark.intel.com/products/78931/Intel-Core-i7-4710MQ-Processor-6M-Cache-up-to-3_50-GHz [retrieved 22 Dec. 2016].

[43] *Intel Atom Processor Z530*, Intel Corp., Santa Clara, CA, http://ark.intel.com/products/35463/Intel-Atom-Processor-Z530-512K-Cache-1_60-GHz-533-MHz-FSB [retrieved 22 Dec. 2016].

[44] Bevilacqua, R., Hall, J. S., Horning, J., and Romano, M., "Ad Hoc Wireless Networking and Shared Computation for Autonomous Multirobot Systems," *Journal of Aerospace Computing, Information, and Communication*, Vol. 6, No. 5, 2009, pp. 328–353. doi:10.2514/1.40734

[45] Arthur, S., Emde, C., and Mc Guire, N., "Assessment of the Realtime Preemption Patches (RT-Preempt) and Their Impact on the General Purpose Performance of the System," *9th Real-Time Linux Workshop*, 2007.

[46] Cerqueira, F., and Brandenburg, B., "A Comparison of Scheduling Latency in Linux, PREEMPT-RT, and LITMUS RT," *9th Annual Workshop on Operating Systems Platforms for Embedded Real-Time Applications*, Paris, France, July 2013, pp. 19–29, http://ertl.jp/~shinpei/conf/ospert13/OSPERT13_proceedings_web.pdf.

[47] Godard, S., *sysstat* [online database], 2016, https://github.com/sysstat/sysstat [retrieved 29 April 2016].

[48] Sojka, M., Pša, P., Lisový, R., Waszniowski, L., and Hamáček, L., "Linux Target for Simulink® Embedded Coder®," 2016, http://lintarget.sourceforge.net [retrieved 29 April 2016].

[49] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T., *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*, Wiley, New York, 2004, Chaps. 1.4, 5–7.

[50] Simon, D., *Optimal State Estimation: Kalman, H-Infinity, and Nonlinear Approaches*, Wiley, New York, 2006, Chaps. 4–7.

[51] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, 3rd ed., Microcosm Press and Springer, New York, 2007, Chap. 10.

[52] Sorenso, H., and Sacks, J., "Recursive Fading Memory Filtering," *Information Sciences*, Vol. 3, No. 2, 1971, pp. 101–119. doi:10.1016/S0020-0255(71)80001-4

[53] Wie, B., *Space Vehicle Dynamics and Control*, 2nd ed., AIAA Education Series, AIAA, Reston, VA, 2008, pp. 480–482, 558–568.

[54] Williams, T. G. C., and Kacur, J., *Cyclictest* [online database], 2016, https://rt.wiki.kernel.org/index.php/Cyclictest> [retrieved 29 April 2016].

[55] "IEEE Standard Specification Format Guide and Test Procedure for Single-Axis Interferometric Fiber Optic Gyros," IEEE Paper STD 952-1997, Piscataway, NJ, 1998. doi:10.1109/IEEESTD.1998.86153

[56] Ng, L. C., and Pines, D. J., "Characterization of Ring Laser Gyro Performance Using the Allan Variance Method," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 1, 1997, pp. 211–214. doi:10.2514/2.4026

[57] Muñoz, J. D., "Rapid Path-Planning Algorithms for Autonomous Proximity Operations of Satellites," Ph.D. Thesis, Univ. of Florida, Gainesville, FL, 2011.

[58] Muñoz, J. D., and Fitz-Coy, N. G., "Rapid Path-Planning Options for Autonomous Proximity Operations of Spacecraft," *AIAA/AAS Astrodynamics Specialist Conference*, AIAA Paper 2010-7667, 2010.

D. A. Spencer
Associate Editor