Journal of Spacecraft and Rockets
Vol. 46, No. 6, November–December 2009

Flight Testing of Multiple-Spacecraft Control on SPHERES During Close-Proximity Operations

Shawn B. McCamish[*] and Marcello Romano[†]
U.S. Naval Postgraduate School, Monterey, California 93943
and
Simon Nolet,[‡] Christine M. Edwards,[§] and David W. Miller[¶]
Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

DOI: 10.2514/1.43563

A multiple-spacecraft close-proximity control algorithm was implemented and tested with the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) facility onboard the International Space Station. During flight testing, a chaser satellite successfully approached a virtual target satellite while avoiding collision with a virtual obstacle satellite. This research contributes to the control of multiple spacecraft for emerging missions, which may require simultaneous gathering, rendezvous, and docking. The unique control algorithm was developed at the U.S. Naval Postgraduate School and integrated onto the Massachusetts Institute of Technology's SPHERES facility. The control algorithm implemented combines the efficiency of the linear quadratic regulator (used for attraction toward goal positions) and the robust collision-avoidance capability of the artificial potential field method (used for repulsion from moving obstacles). The amalgamation of these two control methods into a multiple-spacecraft close-proximity control algorithm yielded promising results, as demonstrated by simulations. Comprehensive simulation evaluation enabled implementation and ground testing of the spacecraft control algorithm on the SPHERES facility. Successful ground testing led to the execution of flight experiments onboard the International Space Station, which demonstrated the proposed algorithm in a microgravity environment.

Nomenclature

A, B, C	=	state-space matrices
$\mathbf{a}$	=	acceleration due to linear-quadratic-regulator- and artificial-potential-field-determined control effort
$\mathbf{a}_{\mathrm{APF}}$	=	acceleration due to artificial-potential-field-determined control effort
$\mathbf{a}_{\mathrm{LQR}}$	=	acceleration due to linear-quadratic-regulator-determined control effort
a_m	=	maximum acceleration
$\mathbf{a}_{\mathrm{obs}}$	=	acceleration of chaser spacecraft toward an obstacle
$a_{x,y,z}$	=	acceleration due to the control effort
D_o	=	obstacle region of influence
d_a	=	goal acceleration decay constant
d_g	=	goal exponential decay constant
d_o	=	stopping distance constant
J_{LQR}	=	linear quadratic regulator cost function
K_{LQR}	=	linear quadratic regulator state feedback gain
k_a	=	acceleration shaping parameter
k_d	=	docking safety parameter
k_g	=	velocity shaping function
k_o	=	obstacle function
k_s	=	safety function
k_v	=	velocity shaping parameter
L_o	=	obstacle exterior surface
N	=	linear quadratic regulator gain matrix
Q	=	linear quadratic regulator state gain matrix
R	=	linear quadratic regulator control effort gain matrix
r	=	Euclidean norm distance or relative range
$\mathbf{r}$	=	relative distance vector
$\mathbf{r}_c$	=	position vector of the chaser spacecraft
$\mathbf{r}_g$	=	position vector of the chaser spacecraft from the goal
r_{init}	=	initial distance of the chaser spacecraft from the goal
r_m	=	maximum allowable distance of the chaser spacecraft from the goal
$\mathbf{r}_o$	=	position vector of the chaser spacecraft from the obstacle
$\mathbf{r}_t$	=	position vector of the target spacecraft with respect to the Earth
S	=	solution of the Riccati equation
$\mathbf{u}$	=	control effort vector
V	=	potential function
V_g	=	goal potential function
V_o	=	obstacle potential function
v_m	=	maximum relative velocity
$\mathbf{v}_o$	=	desired velocity of chaser spacecraft toward an obstacle
$\mathbf{v}_{\mathrm{obs}}$	=	velocity of chaser spacecraft toward an obstacle
$\mathbf{x}$	=	state vector
x, y, z	=	positions, or states, along the Cartesian axis
α_Q	=	linear quadratic regulator state performance gain
β_R	=	linear quadratic regulator control effort gain
Δt	=	time increment
σ	=	standard deviation for the obstacle's region of influence
ω	=	orbital angular velocity

Presented as Paper 6664 at the AIAA Guidance, Navigation, and Control Conference and Exhibit, Honolulu, HI, 18–21 August 2008; received 2 February 2009; revision received 31 July 2009; accepted for publication 3 August 2009.

[*]Lieutenant Colonel, U.S. Air Force; Postdoctoral Researcher, Electrical and Computer Engineering Department, 700 Dyer Road; shanomar1@yahoo.com. Member AIAA.

[†]Assistant Professor, Mechanical and Astronautical Engineering Department, 700 Dyer Road; mromano@nps.edu. Senior Member AIAA.

[‡]Postdoctoral Associate, Space Systems Laboratory, 77 Massachusetts Avenue; snolet@alum.mit.edu. Member AIAA.

[§]Research Assistant, Space Systems Laboratory, 77 Massachusetts Avenue; cmedwards@alum.mit.edu. Member AIAA.

[¶]Professor, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue; millerd@mit.edu. Senior Member AIAA.

I. Introduction

THIS paper presents the simulation, ground verification, and flight-test results of a unique control algorithm for a multiple-spacecraft close-proximity docking experiment [1]. An autonomous distributed spacecraft control algorithm was implemented for a three-satellite test configuration. The algorithm, developed at the U.S. Naval Postgraduate School (NPS), allows for robust collision avoidance of both moving and stationary obstacles while converging efficiently to a desired position. The promising simulation results led to the integration of the NPS control algorithm [2–5] into the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) facility, developed by the Massachusetts Institute of Technology Space Systems Laboratory (MIT SSL) and currently flying onboard the International Space Station (ISS) [6–9]. First, a three-satellite experiment that requires collision avoidance during a docking maneuver was selected. Second, the NPS simulations were modified to include the SPHERES physical characteristics and constraints as well as physical modeling for animations. Third, the NPS control algorithm was integrated into MIT's SPHERES simulation. The NPS and MIT simulations of the spacecraft control algorithm are comparable, with known dissimilarities. Next, the control algorithm was implemented onto the MIT SSL SPHERES ground facility for verification before flight. The positive ground test results allowed all necessary code to be uplinked to the SPHERES facility onboard the ISS. Finally, a flight test was successfully conducted.

Although there exists a significant amount of literature covering the different aspects of autonomous docking [10,11] as well as past autonomous docking missions [12–18], no attempts have been made to date to perform an autonomous docking on orbit while avoiding an obstacle. This paper presents the first of such an attempt. It covers both the theory behind the control algorithm used and the process followed to implement it. More specifically, it outlines the development, simulation, and testing (ground and flight) of a multiple-spacecraft close-proximity control algorithm on the SPHERES facility. Section II describes the NPS control algorithm. Section III presents a quick overview of the SPHERES facility. Then, in Secs. IV and V, the simulations are discussed and the ground verification on the hardware is covered. Finally, the flight-test results are reported in Sec. VI.

II. Overview of NPS Multiple-Spacecraft Close-Proximity Control Algorithm

Simultaneous autonomous control of multiple spacecraft maneuvering will be required for several planned space missions in the near future [19,20]. Large-spacecraft formation tracking and station keeping has received a great deal of study, but research in the area of multiple-spacecraft close-proximity operations is limited [21,22]. There are numerous mission scenarios that involve the divergence or convergence of multiple spacecraft in close proximity [23–25].[**] Present close-proximity path planning and tracking algorithms are computationally expensive and may require manual backup.

Therefore, a relatively simple automated control algorithm is desired that allows for multiple-spacecraft close-proximity operations. Research and experience with terrestrial-based robots have matured the application of artificial potential field (APF) robotic navigation and control algorithms. The simplicity of the APF-based control algorithms is a good match for spacecraft applications with limited proximity sensors and processing capability. Our research proposed and evaluated a control algorithm that combines the efficiency of a linear quadratic regulator (LQR) with an APF-based collision-avoidance capability [1–5]. The APF-based collision avoidance relies on relative positions and velocities. The merged LQR/APF control algorithm uses simple goal commands and obstacle sensory data to achieve robust close-proximity performance and establishes a baseline for fuel efficiency while maintaining collision-free maneuvers [1–4].

Critical evaluation of multiple-spacecraft control algorithms requires high-fidelity 6-degree-of-freedom (6-DOF) spacecraft models. Most proposed spacecraft control algorithms have not been fully assessed with realistic spacecraft dynamics, kinematics, and constraints. The spacecraft physical characteristics and actuator constraints must be included to determine if a spacecraft control algorithm is practical and valid. A high-fidelity 6-DOF spacecraft dynamics model with fully developed nonlinear orbital dynamics and kinematics is used. Given the initial relative positions and velocities of the spacecraft, the orbits are propagated by numerical integration. In particular, a fourth-order Runge–Kutta method was used with $\Delta t = 1.0$ s time increment. This conservative 1.0 Hz sampling rate was selected to allow for slow actuation cycles and sensor update rates. The spacecraft model was developed in MATLAB [26] and validated via Satellite Tool Kit [27]. A full overview of the model and simulation developed for the multiple-spacecraft close-proximity control algorithm is discussed in [5].

A. Linear Model of Multiple-Spacecraft Relative Motion

For this research, the fundamental system is a 6-DOF spacecraft orbiting the Earth. The control algorithm employs linearized relative motion equations. Application in numerical simulations is driven by full nonlinear multiple-spacecraft dynamics and kinematics model, including main perturbations.

The Earth-centered inertial coordinate system (ECI) and the local-vertical/local-horizontal coordinate system (RSW), as depicted in Fig. 1, are used to describe the motion dynamics in this section [28]. The local-vertical/local-horizontal coordinate system with respect to the target satellite body frame is depicted in Fig. 2. To establish the equations of motion between spacecraft, we consider one of the spacecraft as the primary spacecraft (target) and all others as secondary spacecraft (chasers). The Hill–Clohessy–Wiltshire linearized equations of relative motion are [29]

$$\ddot{x} - 2(\omega \cdot \dot{y}) - 3(\omega^2 \cdot x) = a_x \tag{1a}$$

$$\ddot{y} + 2(\omega \cdot \dot{x}) = a_y \tag{1b}$$

$$\ddot{z} + (\omega^2 \cdot z) = a_z \tag{1c}$$

where x, y, and z are the components of the relative position vector $\mathbf{r}$ along the RSW coordinate system, ω is the target spacecraft orbital angular velocity, and $a_{x,y,z}$ are the components of acceleration due to the control effort (thrusters). These equations can be written in a general state-space form as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 0 & 2\omega & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 \\ 0 & 0 & -\omega^2 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \tag{2}$$

These linear equations are used for control algorithm design, whereas a more accurate spacecraft dynamic and kinematic model is exploited during NPS numerical simulations [28].

B. NPS Multiple-Spacecraft Close-Proximity Control Algorithm

During development of the autonomous distributed multiple-spacecraft close-proximity control algorithm, global knowledge was assumed to be not available to each spacecraft [21]. A centralized

[**]Data available online at http://www.darpa.mil/orbitalexpress/ [retrieved 29 August 2007].

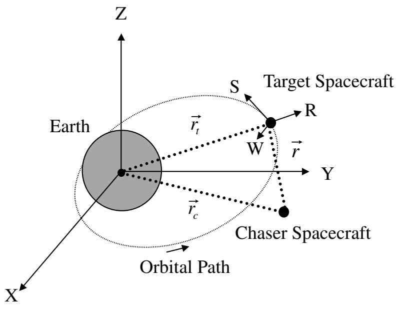


Fig. 1 Relative reference and ECI frame.

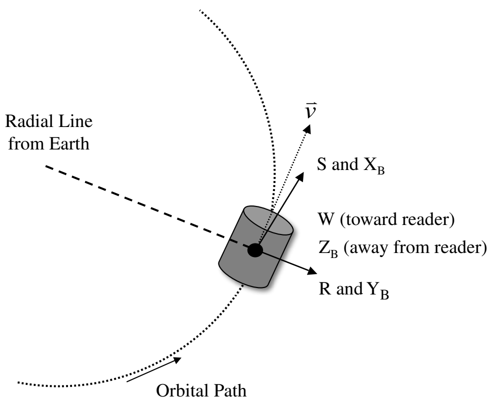


Fig. 2 Relative reference and body frame.

controller was also assumed to not exist, such that each spacecraft must perform its portion of the operation with local information and, possibly, limited communications. The developed control algorithm is a combination of LQR- and APF-based concepts, which use simple goal commands and obstacle sensory data. This control approach is refined and applied to detailed spacecraft dynamics and kinematics models. The spacecraft motion scheme is extended to include collision and obstacle avoidance while conducting close-proximity maneuvers.

Combining the LQR and APF control algorithms results in an efficient and capable amalgamated algorithm. The recursive LQR is used as the attractive force, and APF-based forces, determined by obstacle locations, are used as repulsive forces. For the APF, relative position from goal and obstacles is used to determine the desired velocity. Residuals from the desired velocity are used to command thruster firings. However, the LQR control effort varies with the position and velocity based on the system linearized dynamics. This more complicated relationship requires a modification to both velocity and acceleration in the region of influence of obstacles. The result is an iterative spacecraft control algorithm that is driven by optimal LQR cost convergence, with associated dynamics, and APF-based smooth collision-avoidance responses.

It is important to note that the algorithm presented here is used to control the translational motion of the spacecraft and does not require any specific behavior from the attitude controller other than having the docking port oriented at the target when making contact. In fact, it can be easily integrated with an existing attitude control system, as is the case in the examples shown in this paper. Also, because the algorithm outputs a control effort along each axis, the only assumption made regarding the thruster geometry of the spacecraft is to have control authority along any of the three axes, which should be true for a spacecraft with docking capabilities.

1. *Overview of NPS Spacecraft APF Control*

As a first step toward developing a combined LQR and APF control algorithm, an APF-only control algorithm has been developed, as described in this section. This control algorithm uses potential functions in relation to velocity error, as opposed to only position errors, for controlling spacecraft. Advantages of the space environment are that it is relatively obstacle-free and obstacles are of limited size. In addition, obstacles crossing the orbital path will usually be at high enough relative velocity that collision-avoidance maneuvers are not necessary, or possible, for a spacecraft with local sensor information. The APF of each spacecraft is determined by the arithmetic superposition of the goal and all obstacle potential functions in its working area [30]. The overall potential field can serve as the performance surface for the control algorithm of the form

$$V = V_g + V_o \tag{3}$$

where V_g is the attractive potential of the goal point and V_o is the repulsive potential of obstacles. Selection of the potential functions is critical in ensuring smooth potential fields that are stable and provide the desired performance. One strategy is to select quadratic functions based on the desirable convergence characteristics [31]. The desired velocity can converge along the negative potential gradient as the potential decreases to zero. The APF attractive potential development V_g is not presented in this paper, because its function is to be replaced by an LQR controller. References [1–3] provide a discussion of earlier versions of the algorithm in which APF attractive potential was used.

In APF-based control, a goal potential allows for convergence to the goal position; however, an obstacle potential can also be used to avoid collision with other spacecraft and sensed objects. This repulsion potential curve is a smooth function that increases from the boundary of the region of influence to the surface of the obstacle. The obstacle potential can be selected such that the gradient of the obstacle potential determines the desired velocity modification due to obstacle position. The resulting chaser spacecraft's desired velocity modification, based on the repulsive potential gradient away from an obstacle, is of the form

$$\mathbf{v}_o = \nabla V_o = k_g \cdot k_o \cdot k_s \cdot (\mathbf{r}_o / r_o) \tag{4}$$

where k_g is the velocity shaping function, k_o is the obstacle function, k_s is a docking safety function, and r_o is the range of the chaser spacecraft from an obstacle.

First, the velocity shaping function is used to relate the magnitude of the potential function to the desired velocity. The desired nonnegative velocity shaping function was determined to be

$$k_g = (r_{\text{init}}/r_m) \cdot v_m \cdot (1 - e^{-((1/d_g)\cdot(r_m/r_{\text{init}})\cdot(r_g/2))}) \tag{5}$$

where d_g is a positive constant used to shape the exponential decay of the chaser spacecraft's velocity as it approaches the goal position. The selection of k_g determines the convergence of the control algorithm. Large values may allow the algorithm to converge quickly toward the goal position, but could cause oscillations around the actual goal position. Small values ensure slow steady convergence in a damped manner. This is the more desirable of the possible behaviors for multiple-spacecraft convergence.

Next, the obstacle function used to shape an obstacle region is a Gaussian function of the form

$$k_o = \frac{e^{-r_o^2/(2\cdot\sigma^2)} - e^{-D_o^2/(2\cdot\sigma^2)}}{e^{-L_o^2/(2\cdot\sigma^2)} - e^{-D_o^2/(2\cdot\sigma^2)}} \tag{6}$$

where D_o is the obstacle spatial region of influence, L_o is the minimum distance to the obstacle exterior surface, and σ is the standard deviation for the obstacle region of influence. This selection of k_o ensures that the magnitude of ∇V_o equals ∇V_g at the surface of the obstacle. Both D_o and σ are conservatively selected to ensure that the obstacle region of influence is larger than or equal to the actual dimensions of the obstacle to be avoided. The obstacle region of influence is selected to be

$$D_o = d_o \cdot (L_o + \mathbf{v}_{\text{obs}}^2/(4 \cdot a_m)) \tag{7}$$

with a positive stopping distance constant d_o. The first term in Eq. (7) is a safety margin based on the size of the obstacle, and the second term is the minimum stopping distance of the spacecraft. This allows the chaser vehicle velocity and actuation capability to be incorporated into the obstacle region of influence. The standard deviation σ is selected so that the obstacle surface is within one standard deviation as the spacecraft relative velocity approaches zero, such that $\sigma = D_o/3$. This relationship, modified from [4], allows a reasonable safety region around obstacles and a smooth Gaussian repulsive potential function. Numerous other functions could be selected for the obstacle avoidance potential, such as spherical power law and super quadratic functions [32]. However, these functions would require further a priori knowledge of obstacles, which is not assumed in our work.

An obstacle's region of influence may cause a local minimum or saddle point to occur in the area between the obstacle outer region of influence and the surface of the obstacle. The location of this local minimum depends on the obstacles location with respect to the goal position. This local minimum can cause difficulty if the overall potential function is the only driving function for determining control effort. However, the velocity shaping and obstacle functions, k_g and k_o, respectively, allow for velocity damping around regions of concern. This ensures that the chaser spacecraft slows as it approaches the goal position and avoids obstacles. Balancing these parameters allows the goal position to be placed in the center of a spacecraft and the control algorithm to converge to the surface of the target spacecraft. This is vital capability for docking maneuvers.

As numerous spacecraft and obstacles occupy the chaser spacecraft's region, three simple heuristic rules help regulate the chaser spacecraft collision-avoidance motion. First, the chaser spacecraft are only influenced by obstacles within the region of influence. Second, only obstacles that are at equal distance, or closer, to the goal position are allowed to influence the chaser spacecraft. For instance, the spacecraft is looking toward the goal like an automobile on the road that is only concerned with what is ahead of it and on its sides. In most cases, other spacecraft are simply treated the same as obstacles. The third rule is that obstacles that are further away then the chaser's goal location are not allowed to influence the chaser spacecraft. This ensures that other spacecraft simultaneously docking on the far sides of a target spacecraft do not limit convergence. These logical conditions limit the collision-avoidance considerations needed in obstacle-dense environments and are similar to those presented in [4].

Even with this logic, it is still practical to consider the addition of a docking safety function. If multiple spacecraft rendezvous to the exact same goal position, this will result in a staggered convergence. The first chaser spacecraft to arrive converges to the goal position. The next chaser spacecraft has the additive repulsion of the first spacecraft and converges to a radial position further away. Any additional spacecraft will converge to a range slightly further away. This staggered cluster may be a desirable result for spacecraft rallying to an unknown formation, in which additional command maneuvering may need to occur.

However, for multiple-spacecraft simultaneous docking maneuvers, the staggered cluster effect of the additive repulsion is not desired. In this case, the goal location is an actual position on the target spacecraft, and so each chaser spacecraft must be able to achieve their respective docking location. This can be accomplished by employing a safety function k_s, which modifies the desired repulsive velocity between maneuvering spacecraft and obstacles in the vicinity of the goal. This safety parameter allows for collision avoidance while achieving precision convergence to the goal. The safety function k_s is selected to be a decaying exponential of the attractive potential based on the goal position, such as

$$k_s = 1 - e^{-(r_g - L_o)/2} \tag{8}$$

If $k_s = 1$, then $\mathbf{v}_o$ is not being influenced by the goal location. Otherwise, the repulsion due to other spacecraft decays toward zero as the chaser spacecraft reaches the outer bound of the target spacecraft. In this manner, multiple spacecraft are allowed to converge relatively tightly around the target spacecraft. Limitations in the target spacecraft's outer-boundary surface area and local minima due to saddle points may cause some delays for spacecraft that arrive late. This is only an issue for the second wave of arriving spacecraft as the first spacecraft settle into position. It is envisioned that each spacecraft would be commanded to a specific docking port; therefore, clustered convergence is not a typical operational issue.

The desired chaser spacecraft's acceleration due to obstacles is determined from the summation of all obstacle influences, such as

$$\mathbf{a}_{\text{APF}} = \left(\sum_{\text{obs}} \mathbf{v}_o\right) \Big/ \Delta t \tag{9}$$

where the summation is with respect to all obstacles within the sensor range of the chaser spacecraft. Obstacles may be either other spacecraft (additional chaser spacecraft converging toward a goal within the same region) or stationary obstacles in fixed positions relative to the goal location (for instance, solar panels or thruster plume exclusion zones). This is only the obstacle portion of the APF control algorithm, but its development shows the logic used for the overall LQR/APF control algorithm.

2. *Combined LQR/APF Multiple-Spacecraft Close-Proximity Controller*

An iterative LQR controller provides attraction toward the goal. The multiple-spacecraft LQR algorithm uses the linearized state dynamics from Eq. (2). The iterative LQR allows for efficient control effort based on optimal cost for dynamic system states. This LQR-determined control effort is the desired acceleration due to the actuators, $\mathbf{a}_{\text{LQR}}$. The infinite-time LQR quadratic cost function is of the form

$$J_{\text{LQR}} = \frac{1}{2}\int_0^\infty (\mathbf{x}^T Q \mathbf{x} + \mathbf{u}^T R \mathbf{u} + \mathbf{u}^T N \mathbf{x})\, \mathrm{d}t \tag{10}$$

where Q is the state gain matrix, R is the control effort gain matrix, and the gain matrix N is assumed to be zero for simplicity. The infinite-time cost function was selected to allow convergence duration to be guided by the varying optimal gains. Although the process duration is allowed to be infinite, in practicality, the close-proximity maneuver duration is finite. Also, in a majority of close-proximity maneuvers, the final state conditions are zero (near origin), which results in a negligible endpoint cost component. Therefore, in our research, we eliminated the endpoint cost component and adopted the infinite-time LQR cost function.

If the gain matrix N is assumed to be zero, the optimal feedback control is given by the expression

$$\mathbf{u} = -R^{-1} \cdot (B^T \cdot S) \cdot \mathbf{x} = -K_{\text{LQR}} \cdot \mathbf{x} = \mathbf{a}_{\text{LQR}} \tag{11}$$

where K_{LQR} is the optimal state feedback and S is the solution of the Riccati equation. This LQR-determined control effort $\mathbf{u}$ is the desired acceleration due to the actuators, $\mathbf{a}_{\text{LQR}}$. The weighting matrices can be selected to trade off state convergence and control effort efficiency. For relative spacecraft position and velocity states with control effort along each axis, the diagonal LQR gain matrices are chosen to be of the form

$$Q = \operatorname{diag}\left[\frac{\alpha_Q}{(x_{\max})^2}, \frac{\alpha_Q}{(y_{\max})^2}, \frac{\alpha_Q}{(z_{\max})^2}, \frac{\alpha_Q}{(\dot{x}_{\max})^2}, \frac{\alpha_Q}{(\dot{y}_{\max})^2}, \frac{\alpha_Q}{(\dot{z}_{\max})^2}\right] \tag{12}$$

$$R = \operatorname{diag}\left[\frac{\beta_R}{(u_{x\max})^2}, \frac{\beta_R}{(u_{y\max})^2}, \frac{\beta_R}{(u_{z\max})^2}\right] \tag{13}$$

As an initial guess, the gain matrices are typically selected as diagonal matrices with elements' values normalized by the maximum allowable values of states $x_{\max}$, $y_{\max}$, and $z_{\max}$ and maximum control efforts $u_{x\max}$, $u_{y\max}$, and $u_{z\max}$. The selection of diagonal weighting numerator gains α_Q and β_R for Q and R, respectively, can be tuned based on desired convergence characteristics, as shown in simulation results.

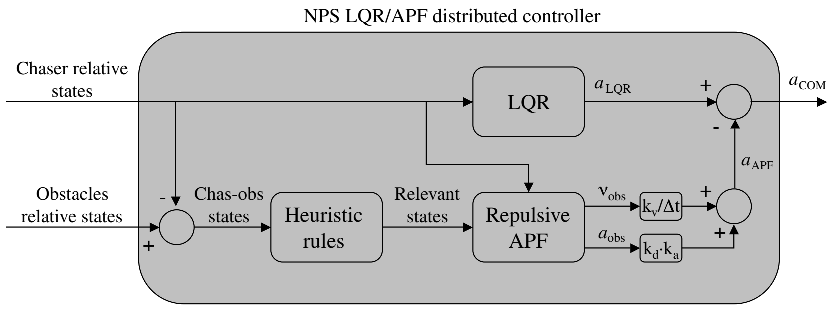


Fig. 3 NPS multiple-spacecraft close-proximity control algorithm summary.

For our research, the LQR gain matrices are selected for efficient control effort and relatively short maneuver duration. During convergence, the cost slope for fixed gain control tends to flatten due to the small state values being considered. This leveling of the cost in the vicinity of the goal can be avoided by using variable gains. Proper gain selection permits steady cost convergence even in the immediate vicinity of the goal. This controller characteristic is essential for submeter spacecraft docking precision.

The LQR state gain matrix scales the chaser spacecraft relative position and velocity as it approaches the goal. The relative position error along each axis is equally weighted by Euclidean norm of the chaser spacecraft position vector from the goal, r_g, by selecting $x_{\max} = y_{\max} = z_{\max} = r_g$ in Eq. (12). Selecting the position gain denominator to be the current distance to the goal allows relative position to become more important as the spacecraft approaches the goal. The relative velocity error along each axis is also equally weighted by selecting $\dot{x}_{\max} = \dot{y}_{\max} = \dot{z}_{\max} = (r_{\text{init}}/r_m) \cdot v_m$ in Eq. (12). This velocity term is determined by scaling the maximum-allowed relative chaser spacecraft velocity v_m by the ratio of the chaser spacecraft's initial range r_{init} and the chaser spacecraft's maximum range r_m. The maximum relative chaser spacecraft velocity should be selected based on available spacecraft actuation and desired maneuver duration. Conservative selection limits the transients due to the initially neutral relative velocity and the convergence rate for safe operations. The numerator terms for the diagonal gains of Eq. (12) are chosen to be $\alpha_Q = r_g$.

The actuator control effort is the acceleration imparted due to translational thrusters. The denominator terms for the diagonal control effort gains in Eq. (13) are set equal to the maximum acceleration, such as $u_{x\max} = u_{y\max} = u_{z\max} = a_m$. The control effort gains are also scaled as the spacecraft relative position changes by selecting $\beta_R = r_g$ in Eq. (13). A minimum scaling factor for the numerator gains can be selected so that as the range to the goal approaches zero, numerical problems and chattering are avoided. For instance, as the r_g approaches zero, the value of r_g is limited to some minimum value, such as $\beta_R \geq 0.05$.

Leveraging the equations developed for APF control, we now apply them to the combined LQR/APF control algorithm. The APF obstacle function represented in Eq. (16) is a Gaussian function, which is equal to 1 at the obstacle boundary. This function will now serve as our LQR/APF velocity shaping parameter due to obstacle position:

$$k_v = k_o = \frac{e^{-r_o^2/(2\cdot\sigma^2)} - e^{-D_o^2/(2\cdot\sigma^2)}}{e^{-L_o^2/(2\cdot\sigma^2)} - e^{-D_o^2/(2\cdot\sigma^2)}} \tag{14}$$

Parameter k_v is multiplied by the component of the chaser spacecraft relative velocity toward an obstacle, $\mathbf{v}_{\text{obs}}$. This ensures that the chaser spacecraft slows to zero at the boundary of the obstacle.

Next, the attractive acceleration due to the LQR/APF recursive function is shaped. There is no change to the LQR when the chaser spacecraft is outside the obstacle regions of influence. However, if the chaser spacecraft is within the region of influence, then acceleration toward the obstacle must be decreased. The LQR/APF acceleration shaping parameter is selected as

$$k_a = e^{-d_a\cdot(r_o - L_o)} \tag{15}$$

where the positive constant d_a is used to establish the parameter's rate of decay. The k_a parameter is multiplied by the component of the chaser spacecraft's desired LQR acceleration in the direction of the obstacle $\mathbf{a}_{\text{obs}}$ to ensure that the LQR/APF does not drive into an obstacle. Finally, the safety function from Eq. (8) is modified to replace the potential function with the chaser spacecraft's range from the goal. The LQR/APF docking safety parameter is given by

$$k_d = 1 - e^{-(d_a\cdot r_g)} \tag{16}$$

The docking safety parameter allows obstacle repulsion to decay faster as the chaser spacecraft approaches the goal position. If the obstacle is the target spacecraft, then the docking safety function allows the chaser to approach in the vicinity of the docking port.

The overall control effort for the multiple-spacecraft LQR/APF with collision avoidance is

$$\mathbf{a} = \mathbf{a}_{\text{LQR}} - \mathbf{a}_{\text{APF}} = \mathbf{a}_{\text{LQR}} - \sum_{\text{obs}}((k_v \cdot \mathbf{v}_{\text{obs}}/\Delta t) + k_d \cdot k_a \cdot \mathbf{a}_{\text{obs}}) \tag{17}$$

with the chaser spacecraft acceleration, due to collision avoidance, determined from the summation of all obstacle influences, such as in Eq. (9).

The control algorithm only decreases velocity and acceleration toward obstacles. It does not actually push away from obstacles. Therefore, densely packed stationary obstacle regions may cause the control to settle into regions other than the goal. However, the relative dynamics result in a force that allows the control algorithm to escape local minimums. The consequence is similar to that achieved by APF wall-following methods [33].

Figure 3 summarizes the LQR/APF algorithm presented in this section. The LQR/APF control is more efficient than the APF control in the absence of obstacles. However, the efficiency gained by LQR/APF control tends to decrease as the number of obstacles in the region increases, because the collision-avoidance capability alters the iterative optimal solution. The addition of the robust collision-avoidance capability is considered to be worth some loss in efficiency in a dense and dynamic obstacle environment. Table 1 summarizes the parameter values used when determining the LQR gain matrices and the shaping functions for the examples in this paper. These values were adapted to the SPHERES testbed in the ISS, which is described in the next section.

Table 1 Parameter values for SPHERES

State gain matrix Q	
r_m	2.0 m
v_m	0.03 m/s
Control effort gain matrix R	
a_m	0.004093 m/s^2
Shaping function parameters	
d_o	3.0 m
L_o	0.306 m
d_a	1.0 m^{-1}

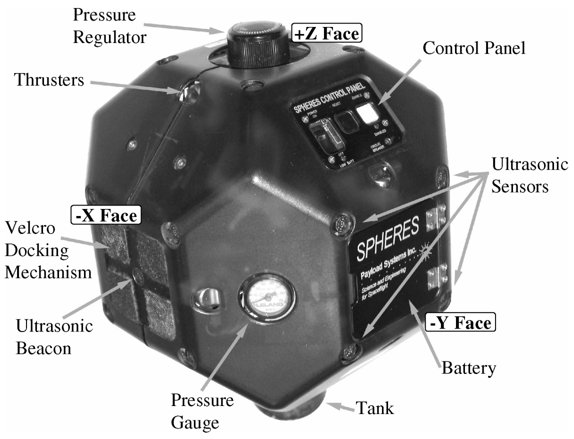


Fig. 4 A SPHERES satellite.

III. Overview of the MIT SPHERES Facility

The MIT SSL has developed a nanosatellite testbed to provide researchers with an experimental laboratory for testing formation flight and autonomous docking algorithms. The SPHERES facility [6–9] consists of three 20-cm-diam nanosatellites (Fig. 4), which can autonomously control their relative positions and orientations in a 6-DOF environment. The testbed is primarily designed to operate inside the ISS, but it can also operate onboard NASA's reduced-gravity aircraft as well as in a two-dimensional environment (3-DOF motion) on a flat floor (such as the one at the NASA Marshall Space Flight Center Flight Robotics Laboratory) or on a laboratory air table [34–36]. By operating inside the ISS, SPHERES exploits the microgravity environment to represent the dynamics of distributed satellite and docking missions while preventing the testbed from experiencing unrecoverable failures when real or simulated guidance, navigation, and control (GNC) failures occur. Therefore, SPHERES provides a fault-tolerant environment to test advanced algorithms and operations.

Each SPHERES satellite is identical and contains most of the subsystems of a standard satellite. It is self-contained with power (AA batteries), propulsion (CO_2 gas), computers, and navigation equipment. The satellites communicate with each other and an ISS laptop through two low-power wireless links. They move in the testing environment by emitting small jets of CO_2 gas using pulse-width-modulated on/off microvalves (thrusters). The navigation system uses time-of-flight data of ultrasonic signals emitted from transmitters at known locations to receivers on the satellites to estimate the satellite's states. More precisely, the basic measurement is the time that an ultrasonic signal takes to travel from a beacon (transmitter) mounted at a known location on the laboratory wall to a microphone (receiver) located on the satellite. Given that there are 5 beacons mounted on the walls and 24 microphones on each satellite, there is a potential of 120 measurements per satellite per measurement event. The time-of-flight data can be converted to range data using the known speed of sound in the ISS. These data combined with data from three gyroscopes are processed using the navigation software module to compute a 6-DOF state solution. The resulting precision on the estimates is a few millimeters in position and approximately 1 deg in attitude in most of the testing volume [37]. When using a control frequency of typically 1 Hz (adjustable), the satellites can maintain a position uncertainty of ± 2 cm inside of the testing volume, even when neglecting the orbital dynamics. Each satellite is also equipped with a Velcro docking mechanism located on its $-X$ face (Fig. 4). The Velcro pattern is identical on each satellite. It tolerates a position error of 2 cm in the plane perpendicular to the docking axis as well as an attitude error of 5 deg, as long as the terminal approach velocity is below 2 mm/s.

The onboard GNC software is written in C. It consists of a series of modules, each accomplishing a specific task (estimation, control, thruster management, and maneuver sequencing) [6,10]. Modularity in the software proved to greatly facilitate integration of algorithms developed by scientists outside MIT [38], such as the case in this paper. It also allows flexibility in designing an experiment through the selection of different combinations of modules. Inheritance between experiments and reuse of previously validated software modules increases the robustness of the code and enables incrementally mature GNC technologies [6].

IV. Simulation Results

A. Three-SPHERES Collision Avoidance and Docking Experiment

The two-spacecraft docking maneuver is the basis for on-orbit servicing and assembly. As multiple spacecraft are required to perform docking maneuvers, several potential complications arise. First, docking of multiple spacecraft requires dedicated docking ports. Second, the docking mechanisms and the docking order need to be addressed. Third, the force and torque tolerance of the docking mechanism and the overall spacecraft need to be considered. Also, the docking mechanisms must be arranged on each spacecraft to allow for sensor fields of view and approach zones.

For spacecraft assembly, the docking order is often predetermined. This is typically the case for heterogeneous spacecraft that must be assembled in a specific order [39]. For homogenous spacecraft, the order of docking may not be as important, but may be limited due to docking mechanism number, position, and function. For instance, a possible cubic spacecraft may dock on any of its six sides. A spacecraft such as SPHERES, with only one docking connection, is more limited in versatility of assembly scenarios.

The selected docking experiment involved the three-SPHERES satellite in the ISS. One satellite serves as the target and another serves as the chaser. The third satellite serves as a floating obstacle, which the chaser must avoid while maneuvering to dock with the target. The limited number of SPHERES satellites and space in the ISS constrained the experiments that could be conducted. The physical characteristics of each SPHERES satellite in the group are assumed to be identical. For simulation purposes, the target satellite altitude is set at the average ISS altitude of 333 km. The initial range between the target and chaser is limited to less than or equal to 2.0 m in RSW coordinates, due to the testing volume dimensions in the ISS. The SPHERES' initial positions, for simulation purposes, are summarized in Table 2. For this experiment, initial velocities of the chaser and the obstacle are assumed to be the same as the target satellite (zero initial relative velocities). The physical characteristics of the SPHERES satellites are approximately 0.268 m in diameter (including the CO_2 tank protrusion) and 4.33 kg in mass (assumed constant because of the small mass of propellant expelled during an experiment). It is interesting to note that the relative proportion of these numbers roughly follows the subsystem sizing guidelines from [40], which makes the SPHERES facility representative of a typical satellite system. The center of mass of the spacecraft is assumed to be located at the geometric center. Translational motion is conducted via thrusters with a maximum thrust of 0.22 N along each of the three primary axes (two thrusters per axis, 0.11 N per thruster). The standard SPHERES attitude controller, a simple proportional–integral–derivative (PID) type of controller described in [6], is used to control the attitude of the spacecraft and ensure the right orientation of the docking port at the time of contact. The chaser

Table 2 SPHERES initial relative positions

Target spacecraft initial position	
R axis	0.00 m
S axis	0.00 m
W axis	0.00 m
Chaser spacecraft initial position	
R axis	0.00 m
S axis	−1.60 m
W axis	0.00 m
Obstacle spacecraft initial position	
R axis	0.05 m
S axis	−0.80 m
W axis	0.00 m

satellite is initially pointing away from the target and has no initial attitude rate.

B. NPS Simulation of the Three-SPHERES Collision Avoidance and Docking Experiment

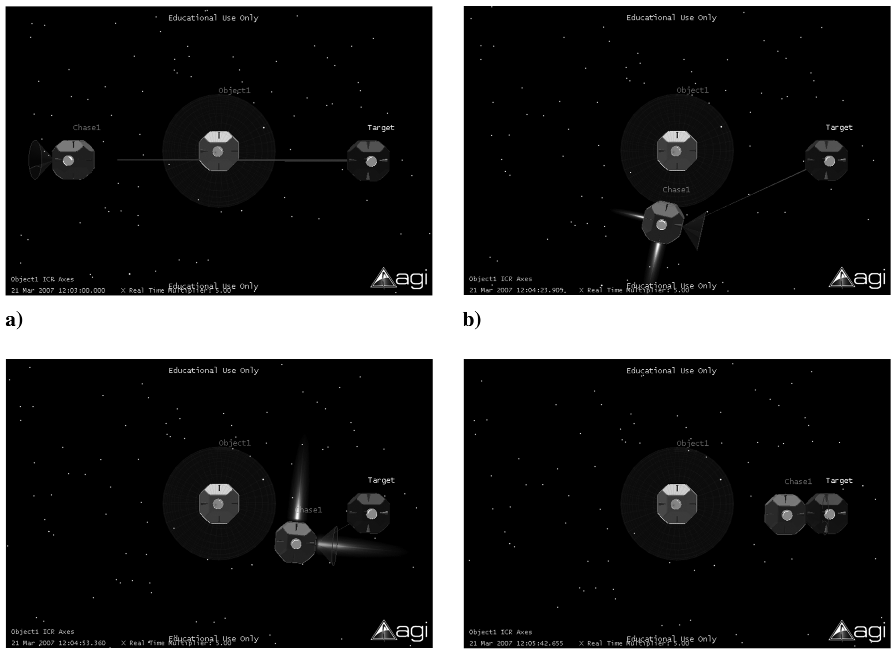
The results of the NPS simulation of the three-SPHERES collision avoidance and docking experiment are presented in this section in the form of an animation of the successful docking maneuver. For this simulation, position and ranging sensors were assumed to provide ideal information. The simulation was animated using Satellite Tool Kit, as discussed in [5]. Frame shots of the animation are shown in Fig. 5. The initial positions and orientations of the satellites are displayed in Fig. 5a, with the target on the right, the chaser on the left, and the obstacle in the middle. The viewpoint is along the W axis of the orbital RSW coordinate frame, with the R axis pointing to the top of the screen and the S axis pointing to the right. The target body axis is oriented with the x body pointing to the right, the y body toward top of the screen, and the z body toward the reader. The chaser has a simulated docking sensor cone protruding from its negative x-body axis. The obstacle is slightly offset along the R axis. It also has a faint mesh wrapped around it to show a general collision-avoidance region of influence. For the purpose of this simulation, it is assumed that the chaser has perfect knowledge of its attitude as well as the one of the target. The next screen shot, Fig. 5b, shows the collision-avoidance maneuver of the chaser, following the edge of the region of influence. The chaser has rotated to align its docking port with the one of the target. The thruster plums are estimated from simple thruster mapping. After avoiding the obstacle, the chaser maneuvers for docking as shown in Fig. 5c. Finally, it docks with the target along the target's negative x-body axis. The final docked positions are shown in Fig. 5d. With a terminal position error of 2 mm in the plane perpendicular to the docking axis and a terminal approach velocity of 0.4 mm/s, the SPHERES position and velocity requirements for successful docking, enumerated in Sec. III, were met. This simulation successfully represented the first level of integration of the NPS control algorithm into the SPHERES facility. The implementation of the algorithm in a simulation with accurate hardware models follows.

C. Implementation in the MIT SPHERES MATLAB Simulation

The MIT SSL has developed a MATLAB simulation environment to help guest scientists implement GNC software for SPHERES experiments. The SPHERES MATLAB simulation combines a MATLAB translation of the C software library implemented on the SPHERES hardware as well as a realistic state propagator that uses a stochastic model of the onboard sensors and actuators. Although it currently does not model computation or communication delays, it is based on the same GNC architecture as on the hardware. It is primarily used as an intermediate step in implementing GNC algorithms on SPHERES and as a software integration and verification tool for 6-DOF operations before uploading the flight code to the ISS [10].

Figure 6 illustrates the GNC architecture used on the SPHERES facility for the chaser spacecraft, both in the MATLAB simulation and on the hardware. The estimation, communication, and control processes are all asynchronous. The LQR/APF algorithm presented here (grayed box in Fig. 6) is executed in parallel of the SPHERES standard attitude controller at a control frequency of 1 Hz. It uses navigation information provided by onboard sensor data (gyroscopes and ultrasonic sensors) processed with an extended Kalman filter [37]. Navigation states of neighboring spacecraft are provided wirelessly through simulated communication to simulate remote sensing capabilities. The subtraction of these states from the chaser states provides the relative navigation information necessary to the LQR/APF algorithm. The forces commanded by the algorithm, combined with torques commanded by the attitude controller, are processed by a pulse-width modulator and converted into thruster on/off times at the control frequency [10,34].

The MIT SPHERES simulation and the NPS general spacecraft simulation have some minor differences. Primarily, because the MIT SPHERES is an actual hardware system, its dimensions, constraints and sensor uncertainties are modeled. The inclusion of these uncertainties into the SPHERES MATLAB simulation enables a more realistic knowledge of both the translation and attitude states. However, the position uncertainty results in a practical limitation of the docking precision. Although the general NPS simulation assuming ideal sensors could dock with a precision of less than 2 mm, the inclusion of modeled uncertainty resulted in a decreased but more realistic docking precision. The docking precision for the MIT simulation was found to be 6 mm, at a terminal approach velocity of 1 mm/s, still well within the SPHERES position and velocity requirements for successful docking. This demonstrates the control algorithm's subcentimeter performance without requiring additional sensor precision. Second, because the SPHERES are floating inside the ISS for short durations, the orbital dynamics and perturbations are



a) b) c) d)

Fig. 5 Simulation results: animation of the three-SPHERES collision avoidance docking experiment.

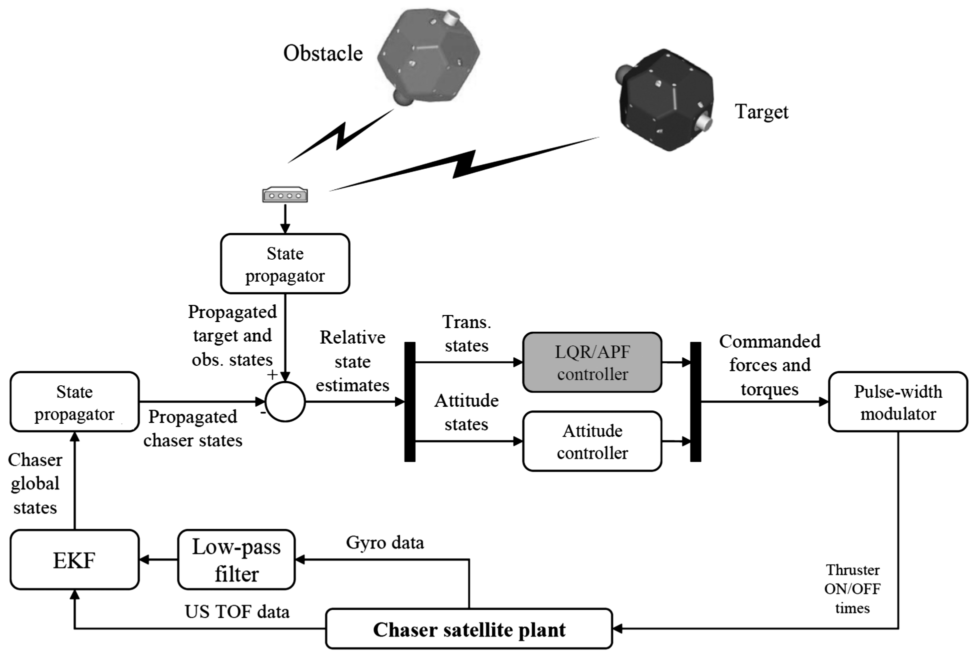


Fig. 6 Guidance, navigation, and control architecture on the SPHERES facility.

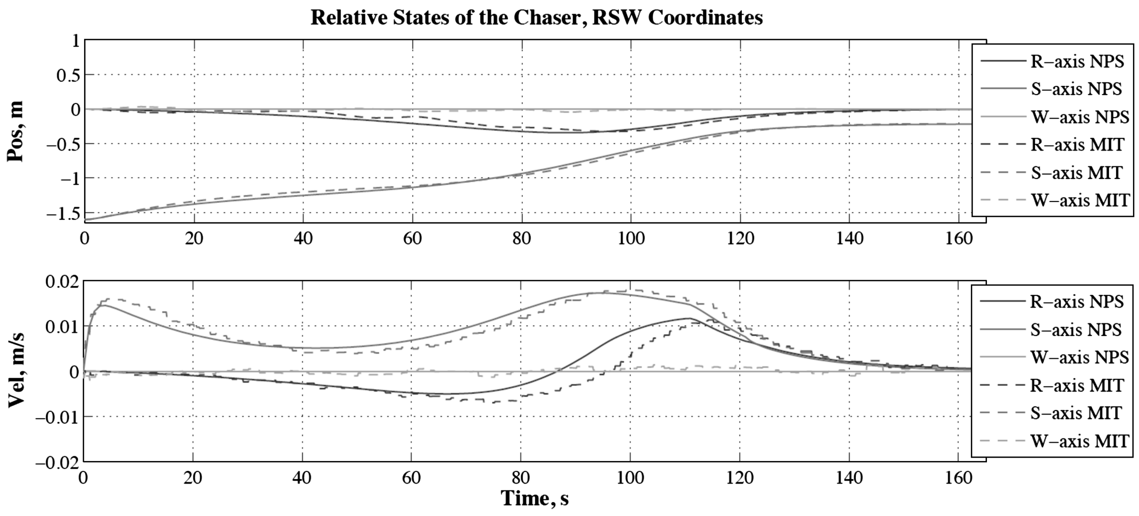


Fig. 7 Simulation results: comparison of the relative states of the chaser calculated using both NPS and MIT simulation tools.

assumed to be negligible. The NPS spacecraft simulation does not negate orbital dynamics and perturbations. Therefore, the MIT simulation includes state uncertainties, and the NPS simulation included orbital dynamics and perturbations.

Before running the SPHERES MATLAB simulations at MIT, the general NPS simulation was adapted to the physical dimensions and dynamic constraints of the SPHERES facility. These adjustments were minor and limited. First, because the flight experiments timelines are limited due to astronauts' schedules, the velocity profile was modified to allow for a test completion within 5 min. This was accomplished by allowing the maximum relative velocity to be $v_m = 0.03\ \mathrm{m/s}$, within the relatively close experimental range. Second, the specific SPHERES satellite physical dimensions replaced the generalized spacecraft characteristics. The MIT and NPS simulations were conducted independently.

A comparison of the results is presented in Fig. 7 for the chaser's relative position and velocity with respect to the local-vertical/local-horizontal coordinate system. The NPS results are solid lines, and the MIT results are shown as dashed lines. The relative differences between the simulations, which could be approximated by a 10 s lag, are attributed to the modeled uncertainties and the absence of direct velocity measurements on the MIT simulations. The results, leading to successful docking in both NPS and MIT simulations, showed that the NPS multiple-spacecraft close-proximity control algorithm is robust to uncertainty in the sensors and actuators and is mature enough to be implemented on the SPHERES hardware.

Table 3 SPHERES initial positions, flight test

Target spacecraft initial position	
R axis	0.00 m
S axis	−0.45 m
W axis	−0.10 m
Chaser spacecraft initial position	
R axis	0.00 m
S axis	0.00 m
W axis	0.00 m
Obstacle spacecraft initial position	
R axis	0.00 m
S axis	0.65 m
W axis	−0.10 m

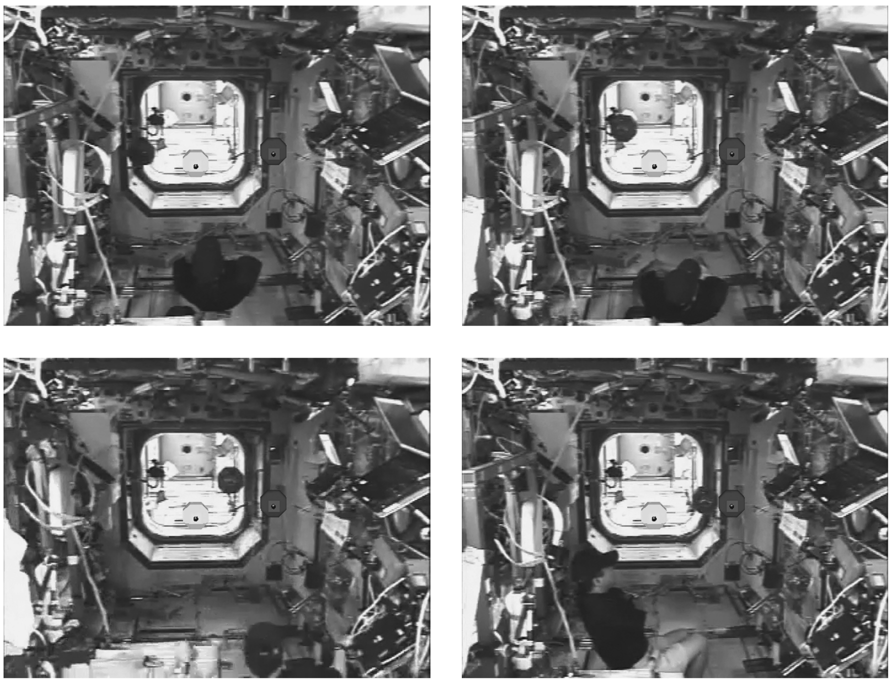


Fig. 8 On-orbit experimental results: single-satellite collision avoidance docking experiment in the ISS, with graphical representation of the virtual obstacle and target satellites.

V. SPHERES Code Verification

There were several steps that were performed to verify that the NPS multiple-spacecraft close-proximity control algorithm was successfully implemented on the SPHERES facility. As discussed in the previous section, after the algorithm from the NPS simulation was implemented in the SPHERES simulation, the behavior of the satellites in both simulations was checked to ensure that they matched closely. Then the controller code from the SPHERES MATLAB simulation was translated into the C programming language, using Microsoft Visual C + + [41] to run on the satellite. The repository for the SPHERES satellite software contains a library that includes the matrix manipulation functions that are used in the NPS controller. Thus, the only additional function that needed to be developed in C was a LQR solver. Using a combination of techniques proposed in [42], a simple LQR solver adapted to a double integrator system was derived for controlling the position of the satellites in all three axes.

The first step to verify that the NPS controller was successfully implemented in C was to compile the C translation into a MEX-file[††] accessible by the SPHERES MATLAB simulator. The resulting trajectories were compared with those of the original NPS simulation. They were found to be practically identical to the originals, demonstrating the accuracy of the C translation of the NPS controller.

The second step was to test the C translation with hardware-in-the-loop to ensure compatibility of the algorithm with the SPHERES real-time operating system. When the NPS controller ran on the SPHERES computer, it required $\sim$10 ms ($\pm$2 ms) per iteration. Because the algorithm would typically only be called at a frequency of 1 Hz, this computation delay was judged to be acceptable for the real-time command structure of SPHERES.

The third step was to run tests on the flat table in the SPHERES laboratory. Because the NPS control algorithm uses very small thrust levels (thruster opening times on the order of 20 ms), the ground satellite had difficulty overcoming the stiction, friction, and slope irregularities of the table. Therefore, docking could not be achieved reliably on the MIT SSL flat table. However, the ground tests were helpful to visually verify that the satellite was thrusting in the correct directions at different moments in the maneuver.

[††]A file consisting of a C program compiled in a format that can be read in MATLAB.

Finally, because the ground tests could not clearly demonstrate that the thrust levels were correct, the telemetry was analyzed after this controller was run briefly on a SPHERES satellite in the ISS in December 2007. The controller inputs from the telemetry were run through the original MATLAB version of the NPS controller, provided by NPS. The resulting thrust levels for each axis matched almost exactly with those read through the telemetry of the flight test in the ISS. This indicated once more that the onboard controller behaves very similarly to the MATLAB NPS controller. From the results of these four verification steps, it was concluded that the controller was successfully implemented on the SPHERES hardware and ready for its final test in the ISS.

VI. SPHERES ISS Flight Testing

On 2 December 2007, 29 December 2007, and 27 January 2008, the NPS test was executed a total of six times as part of the SPHERES test sessions 10, 10a, and 11, all performed onboard the ISS. During the four tests in test session 10 and the one in test session 10a, three SPHERES satellites were used (a chaser, a target, and an obstacle). The satellites did not successfully complete their maneuvers, mainly because of confusion with the deployment instructions sent to the crew. Thus, although valuable information was collected through the telemetry, the experiments performed during these two test sessions did not demonstrate the capabilities of the NPS multiple-spacecraft close-proximity control algorithm.

For test session 11, the on-orbit supply of SPHERES batteries was low, with a new supply scheduled to launch in February 2008. To conserve batteries, the test was modified to use a single satellite, with the location of the target and the obstacle predetermined and hard-coded in the software, to simulate a virtual target and a virtual obstacle. The initial position of each satellite in a coordinate frame attached to the ISS is listed in Table 3. Figure 8 shows a few frames taken from a combination of real footage and an animation of the telemetry collected during the experiment. The chaser is originally on the left, the virtual target on the right, and the virtual obstacle in the center. Although not physically present during the experiment shown in Fig. 8, the virtual target and the virtual obstacle were drawn to better illustrate the trajectory followed by the chaser and to add credibility to the results. The chaser successfully maneuvered around the virtual obstacle and approached the virtual target following a

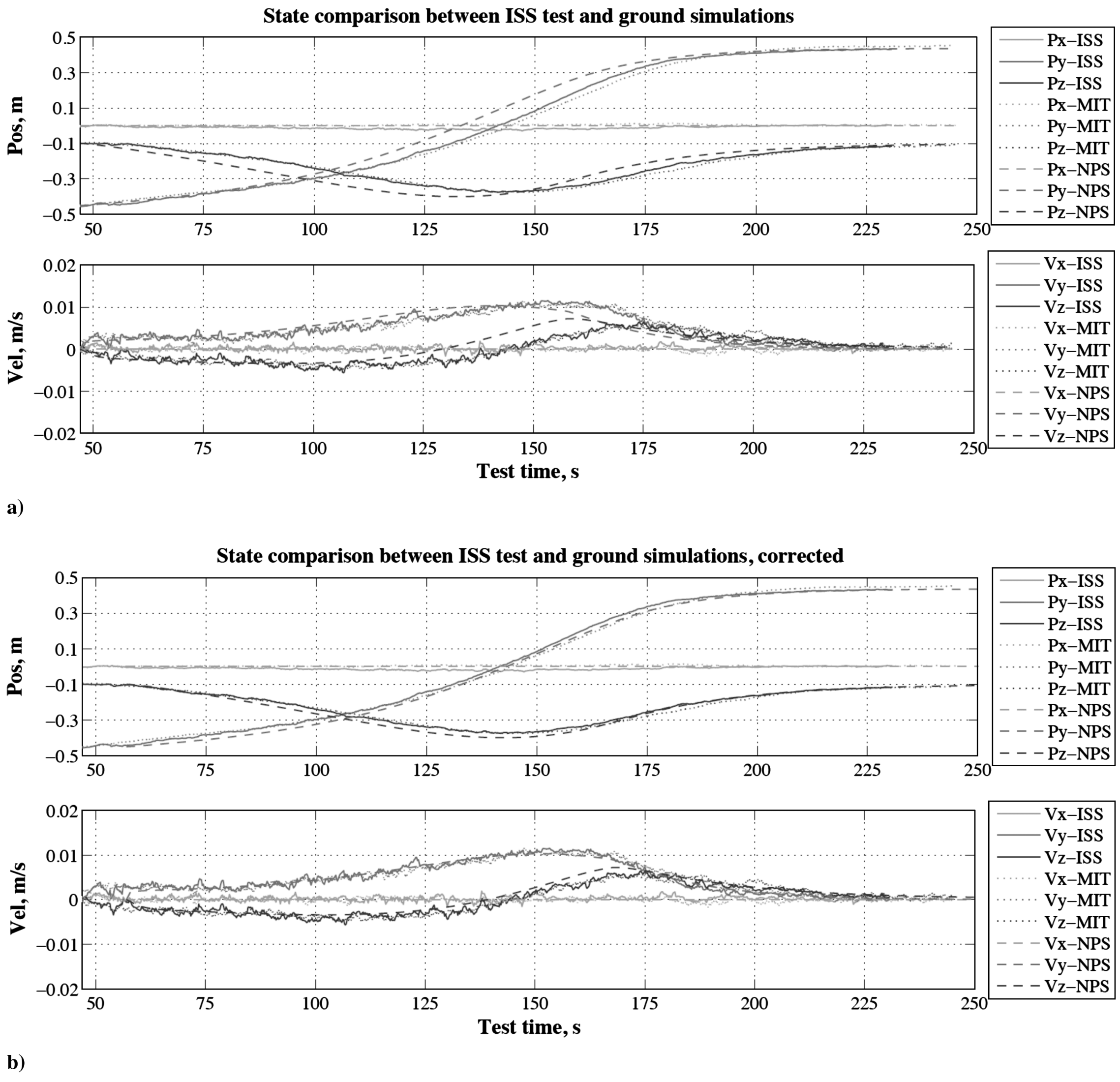


Fig. 9 On-orbit experimental results: position and velocity of the chaser during the single-satellite collision avoidance docking experiment in the ISS.

trajectory that would lead to docking, given a real target at that location.

The telemetry from the ISS test provided the position and velocity of the chaser, which are plotted in Fig. 9a (continuous lines). The origin or the coordinate system is located in the center of the test volume and also corresponds to the geometric center of the obstacle for this experiment. The first 47 s were used by the onboard navigation system to converge to a solution and to move the chaser in a proper initial position, given the constraints of the test volume. The NPS multiple-spacecraft close-proximity control algorithm controlled the chaser from 47 s until the end of the experiment. The trajectory is very smooth throughout the experiment. The apparent noise on the velocity estimates is caused by the derivation of the noisy ultrasonic range measurements. Docking would have occurred at 230 s, with a position error of 1.6 cm in the plane perpendicular to the docking axis. The velocity along all three axes at that time was

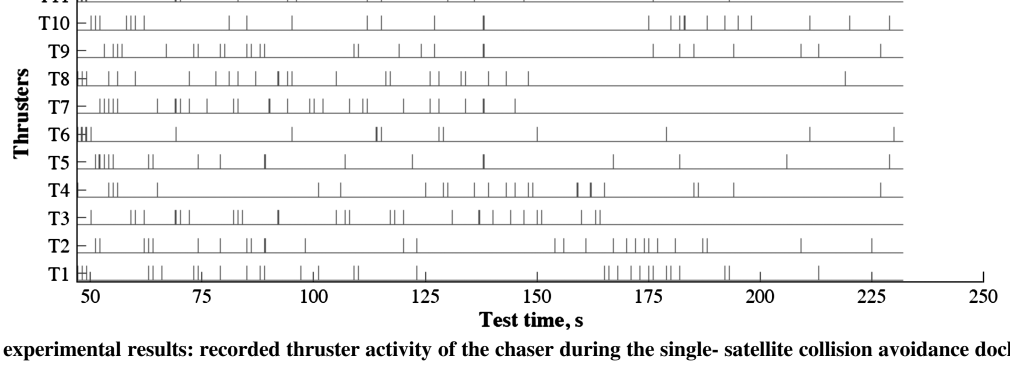


Fig. 10 On-orbit experimental results: recorded thruster activity of the chaser during the single- satellite collision avoidance docking experiment in the ISS.

estimated to be less than 1 mm/s (the approach velocity itself was just around 0 mm/s), which would have resulted in a soft docking. Figure 9a also shows predictions from the NPS and MIT ground simulations with the same initial conditions (dotted lines). As was observed in early simulations (Fig. 7), there is a difference between NPS and MIT simulations, which can be approximated by a 10 s lag, attributable to modeling discrepancies in the simulations and the absence of direct velocity measurements in the MIT simulations. However, when shifting the NPS trajectory profile by 10 s (Fig. 9b), all three trajectory profiles overlap well. This experiment proved the success of the implementation of the NPS collision-avoidance docking on the SPHERES facility.

Figure 10 shows the thruster activity downloaded in the telemetry during the docking phase of the experiment. A higher concentration of thruster firings is observed at the beginning of the docking phase, between 47 and 55 s, to initiate motion. However, as the chaser gets closer to the target, it gradually reduces its approach velocity in such a way that firings are minimal close to docking.

Overall, with the SPHERES position and velocity requirements for successful docking being met, this experiment was very successful. Not considering plume impingement perturbations and assuming accurate position and velocity control of the target satellite, it is fair to say that the trajectory profile obtained in this experiment would have resulted in docking given a target satellite physically present at the location at which the virtual target was set.

VII. Conclusions

The NPS multiple-spacecraft close-proximity control algorithm was successfully implemented on the MIT SSL SPHERES facility and tested onboard the ISS. The algorithm, which combines LQR efficiency for attraction toward goal positions with APF-based collision avoidance for repulsion from moving obstacles, performed docking while simultaneously avoiding an obstacle. This paper presented the algorithm itself as well as the process followed to implement, verify, and demonstrate it in microgravity on the SPHERES facility. Two series of simulations performed independently, one at NPS and the other one at MIT, provided incremental levels of algorithm integration. Then SPHERES ground testing supported the flight experiment through successful verification of the flight software on the integrated system. Finally, a successful flight experiment demonstrated docking with a virtual target while avoiding a virtual obstacle. By allowing docking in a microgravity environment, the algorithm presented here contributes to the control of multiple spacecraft during close-proximity operations.

Acknowledgments

The authors would like to acknowledge Riccardo Bevilacqua from the U.S. Naval Postgraduate School and Alvar Saenz-Otero from Massachusetts Institute of Technology for their support on the project. Furthermore, the authors would like to thank Dan Tani of NASA for having operated the SPHERES on the International Space Station (ISS). The authors are also very grateful to the U.S. Department of Defense and their Space Test Program for providing the opportunity to perform experiments in the ISS. This research was partially funded by the U.S. Department of Defense.

References

[1] McCamish, S. B., Romano, M., Nolet, S., Edwards, C. M., and Miller, D. W., "Ground and Space Testing of Multiple Spacecraft Control During Close-Proximity Operations," AIAA Paper 2008-6664, Aug. 2008.

[2] McCamish, S. B., "Distributed Autonomous Control of Multiple Spacecraft During Close Proximity Operations," Ph.D. Dissertation, Electrical Engineering Dept., U.S. Naval Postgraduate School, Monterey, CA, 2007.

[3] McCamish, S. B., Romano, M., and Yun, X., "Autonomous Distributed Control Algorithm for Multiple Spacecraft in Close Proximity Operations," AIAA Paper 2007-6857, Aug. 2007.

[4] McCamish, S. B., Romano, M., and Yun, X., "Autonomous Distributed LQR/APF Control Algorithm for Multiple Small Spacecraft During Simultaneous Close Proximity Operations," *Proceedings of the 21st Annual AIAA/USU Conference on Small Satellites* [CD-ROM], AIAA, Reston, VA, Aug. 2007.

[5] McCamish, S. B., and Romano, M., "Simulations of Relative Multiple-Spacecraft Dynamics and Control by MATLAB-Simulink and Satellite Tool Kit," AIAA Paper 2007-6805, Aug. 2007.

[6] Nolet, S., Saenz-Otero, A., Miller, D. W., and Fejzic, A., "SPHERES Operations Aboard the ISS: Maturation of GN&C Algorithms in Microgravity," *Proceedings of the 30th Annual AAS Rocky Mountain Guidance & Control Conference*, American Astronautical Society, Breckenridge, CO, 2007, pp. 247–266; also American Astronautical Society, Paper 07-042, Feb. 2007.

[7] Saenz-Otero, A., Aoude, G., Jeffrey, M. M., Mohan, S., Fejzic, A., Katz, J., Edwards, C. M., and Miller, D. W., "Distributed Satellite Systems Algorithm Maturation with SPHERES Aboard the ISS," 59th International Astronautical Congress, Glasgow, Scotland, U.K., International Astronautical Congress, Paper IAC-08-A.2.6.B4, Sept.–Oct. 2008.

[8] Kong, E. M. C., Otero, A. S., Nolet, S., Berkovitz, D. S., Miller, D. W., and Sell, S. W., "SPHERES as a Formation Flight Algorithm Development and Validation Testbed: Current Progress and Beyond," *2nd International Symposium on Formation Flying Missions and Technologies*, NASA CP-2005-212781, Sept. 2004.

[9] Kong, E. M., Hilstad, M. O., Nolet, S., and Miller, D. W., "Development and Verification of Algorithms for Spacecraft Formation Flight Using the SPHERES Testbed: Application to TPF," *New Frontiers in Stellar Interferometry*, edited by W. A. Traub, Proceedings of SPIE, Vol. 5491, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2004, pp. 308–319.
doi:10.1117/12.552339

[10] Nolet, S., "Development of a Guidance, Navigation and Control Architecture and Validation Process Enabling Autonomous Docking to a Tumbling Satellite," Sc.D. Dissertation, Dept. of Aeronautics and Astronautics, Massachusetts Inst. of Technology, Cambridge, MA, June 2007.

[11] Fehse, W., *Automated Rendezvous and Docking of Spacecraft*, Cambridge Aerospace Series, Vol. 16, Cambridge Univ. Press, Cambridge, England, U.K., 2003.

[12] Oda, M., "Experiences and Lessons Learned from the ETS-VII Robot Satellite," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '00)*, Vol. 1, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, April 2000, pp. 914–919.

[13] Kawano, I., Mokuno, M., Kasai, T., and Suzuki, T., "Result and Evaluation of Autonomous Rendezvous Docking Experiments of ETS-VII," AIAA Paper 1999-4073, Aug. 1999.

[14] Davis, T. M., and Melanson, D., "XSS-10 MicroSatellite Flight Demonstration Program Results," *Spacecraft Platforms and Infrastructure*, edited by P. Tchoryk Jr., and M. Wright, Proceedings of SPIE, Vol. 5419, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2004, pp. 16–25.
doi:10.1117/12.544316

[15] Zimpfer, D., Kachmar, P., and Tuohy, S., "Autonomous Rendezvous, Capture and In-Space Assembly: Past, Present and Future," AIAA Paper 2005-2523, Jan. 2005.

[16] Rumford, T. E., "Demonstration of Autonomous Rendezvous Technology (DART) Project Summary," *Space Systems Technology and Operations*, edited by P. Tchoryk Jr., and J. Shoemaker, Proceedings of SPIE, Vol. 5088, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2003, pp. 10–19.
doi:10.1117/12.498811

[17] Shoemaker, J., and Wright, M., "Orbital Express Space Operations Architecture Program," *Spacecraft Platforms and Infrastructure*, edited by P. Tchoryk Jr., and M. Wright, Proceedings of SPIE, Vol. 5419, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2004, pp. 57–65.
doi:10.1117/12.544067

[18] Delage, R., Durante, N., Wasbauer, J. J., Goester, J. F., and Cornier, D., "An Overview of ATV Integrated Mission Analysis and Mission Preparation," 54th International Astronautical Congress, Bremen, Germany, International Astronautical Federation, Paper IAC-03-V.2.09, Sept. 2003.

[19] Mathieu, C., and Weigel, A. L., "Assessing the Flexibility Provided by Fractionated Spacecraft," AIAA Paper 2005-6700, Aug. 2005.

[20] Brown, O., Ermenko, P., and Roberts, C., "Cost Benefit Analysis of a Notational Fractionated SATCOM Architecture," AIAA Paper 2006-5328, June 2006.

[21] Chen, Y. Q., and Wang, Z., "Formation Control: a Review and a New Consideration," *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Inst. of Electrical and

Electronics Engineers, Piscataway, NJ, 2005, pp. 3181–3186.

[22] Ren, W., and Beard, R. W., "Formation Feedback Control for Multiple Spacecraft via Virtual Structures," *IEE Proceedings. Control Theory and Applications*, Vol. 151, No. 3, May 2004, pp. 357–368. doi:10.1049/ip-cta:20040484

[23] David, L., "Military Micro-Sat Explores Space Inspection, Servicing Technologies," *Space.com* [online journal], 22 July 2005, http://www.space.com/businesstechnology/050722_XSS-11_test.html [retrieved 30 July 2009].

[24] Davis, T. M., Baker, T. L., Belchak, T. T., and Larsen, W. R., "XSS-10 MicroSatellite Flight Demonstration Program," *Proceedings of the 17th Annual AIAA/USU Conference on Small Satellites* [CD-ROM], AIAA, Reston, VA, Aug. 2003.

[25] Dornheim, M. A., "Orbital Express to Test Full Autonomy for On-Orbit Service," *Aviation Week & Space Technology*, Vol. 164, No. 23, June 2006, pp. 46–50.

[26] MATLAB, Software Package, Ver. 7.3.0.267 (R2006b), The MathWorks, Inc., Natick, MA, Aug. 2006.

[27] Satellite Tool Kit, Software Package, Ver. 8.1.3 Basic Edition, Analytical Graphics, Inc., Exton, PA, Mar. 2008.

[28] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, 2nd ed., Microcosm Press, El Segundo, CA, 2001.

[29] Clohessy, W. H., and Wiltshire, R. S., "Terminal Guidance System for Satellite Rendezvous," *Journal of the Aerospace Sciences*, Vol. 27, No. 9, 1960, pp. 653–658.

[30] Newman, W. S., and Hogan, N., "High Speed Robot Control and Obstacle Avoidance Using Dynamic Potential Functions," *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 4, Inst of Electrical and Electronics Engineers, Piscataway, NJ, March 1987, pp. 14–24.

[31] Khalil, H. K., *Nonlinear Systems*, 3rd ed., Prentice–Hall, Upper Saddle River, NJ, 2002.

[32] McQuade, F., "Autonomous Control for On-Orbit Assembly Using Artificial Potential Functions," Ph.D. Dissertation, Dept. of Aerospace Engineering, Univ. of Glasgow, Glasgow, Scotland, U.K., 1997.

[33] Yun, X., and Tan, K., "A Wall-Following Method for Escaping Local Minima in Potential Field Based Motion Planning," *Proceedings of the 1997 8th International Conference on Advanced Robotics (ICAR '97)*, Inst of Electrical and Electronics Engineers, Piscataway, NJ, 1997, pp. 421–426.

[34] Nolet, S., Kong, E., and Miller, D. W., "Autonomous Docking Algorithm Development and Experimentation Using the SPHERES Testbed," *Spacecraft Platforms and Infrastructure*, edited by P. Tchoryk Jr., and M. Wright, Proceedings of SPIE, Vol. 5419, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2004, pp. 1–15. doi:10.1117/12.547430

[35] Nolet, S., Kong, E., and Miller, D. W., "Design of an Algorithm for Autonomous Docking with a Freely Tumbling Target," *Modeling, Simulation, and Verification of Space-Based Systems II*, edited by P. Motaghedi, Proceedings of SPIE, Vol. 5799, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2005, pp. 123–134. doi:10.1117/12.603178

[36] Nolet, S., and Miller, D. W., "Autonomous Docking Experiments Using the SPHERES Testbed Inside the ISS," *Sensors and Systems for Space Applications*, edited by R. T. Howard, and R. D. Richards, Proceedings of SPIE, Vol. 6555, SPIE—The International Society for Optical Engineering, Bellingham, WA, 2007, pp. 1–2. doi:10.1117/12.720125

[37] Nolet, S., "The SPHERES Navigation System: From Early Development to On-Orbit Testing," AIAA Paper 2007-6354, Aug. 2007.

[38] Enright, J., Hilstad, M., Saenz-Otero, A., and Miller, D., "The SPHERES Guest Scientist Program: Collaborative Science on the ISS," *Proceedings of the 2004 IEEE Aerospace Conference*, Vol. 1, Inst of Electrical and Electronics Engineers, Piscataway, NJ, 2004, pp. 35–56.

[39] Yang, G., Yang, Q., Kapila, V., Palmer, D., and Vaidyanathan, R., "Fuel Optimal Manoeuvres for Multiple Spacecraft Formation Reconfiguration Using Multi-Agent Optimization," *International Journal of Robust and Nonlinear Control*, Vol. 12, Feb.–March 2002, pp. 243–283. doi:10.1002/rnc.684

[40] Larson, W. J., and Wertz, J. R., *Space Mission Analysis and Design*, 3rd ed., Microcosm Press, El Segundo, CA, 2004.

[41] Microsoft Visual C + +, Software Package, Ver. 6.0, Microsoft Corp., Cleveland, OH, June 1998.

[42] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes in C, The Art of Scientific Computing*, 2nd ed., Cambridge Univ. Press, Cambridge, England, U.K., 1992.

C. McLaughlin
Associate Editor