JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 29, No. 6, November–December 2006

Linear Covariance Techniques for Orbital Rendezvous Analysis and Autonomous Onboard Mission Planning

David K. Geller*
Utah State University, Logan, Utah 84322

A novel trajectory control and navigation analysis software approach is developed. The program quickly determines trajectory dispersions, navigation errors, and required maneuver Δv at selected key points along a nominal trajectory. It can be used for mission design and planning activities or in autonomous flight systems to help determine the best trajectories, the best maneuver locations, and the best navigation update times to ensure mission success. These features are illustrated with two simple examples. The software works by applying linear covariance analysis techniques to a closed-loop guidance, navigation, and control (GN&C) system. The nonlinear dynamics and flight software models of a closed-loop six-degree-of-freedom Monte Carlo simulation are linearized. Then, linear covariance techniques are used to produce a program that will accurately predict 3-σ trajectory dispersions, navigation errors, and Δv variations. Although the application presented is orbital rendezvous, the tools, techniques, and mathematical formulations are applicable to a variety of other space missions and GN&C problems including atmospheric entry, low-thrust electric propulsion missions, translunar injection and lunar orbit insertion, lunar ascent/descent, and formation-flying missions.

Nomenclature

$\mathcal{C}_A$ = covariance of augmented state vector
D_{nav} = covariance of navigation state dispersions
D_{true} = covariance of true state dispersions
$\text{Diag}(\boldsymbol{v})$ = diagonal matrix with elements of vector $\boldsymbol{v}$ along the diagonal
$\boldsymbol{F}^i_{\text{aero}}$ = aerodynamic force in an inertial coordinate frame
$\boldsymbol{F}^i_{\text{grav}}$ = gravitational force in an inertial coordinate frame
$I_{n\times n}$ = identity matrix
$O_{m\times n}$ = matrix of zeros
P_{true} = true covariance of navigation state error
$\hat{P}$ = flight computer covariance of navigation state error
$\boldsymbol{p}_{\text{aero}}$ = aerodynamic model parameters
$\boldsymbol{p}_{\text{atm}}$ = atmosphere model parameters
$\boldsymbol{p}_{\text{gyro}}$ = gyro model parameters
$\boldsymbol{p}_{\text{optrk}}$ = optical tracking camera model parameters
$\boldsymbol{p}_{\text{starcam}}$ = star-camera model parameters
$\boldsymbol{p}_{\text{torque}}$ = torque actuator model parameters
$\boldsymbol{p}_{\Delta v}$ = impulsive actuator model parameters
$\boldsymbol{q}$ = quaternion indicated by boldface q
$\boldsymbol{q}^a_b$ = quaternion representing the orientation of the b frame with respect to the a frame
$\boldsymbol{q}^i_c$ = quaternion representing the orientation of the chaser frame with respect to the inertial frame
$\boldsymbol{q}^i_o$ = quaternion representing the orientation of the object frame with respect to the inertial frame
$\boldsymbol{R}^c_{\text{attach}}$ = position the attach point in the chaser body frame
$\boldsymbol{R}^d_{\text{des}}$ = desired position of chaser attach point relative to the object docking port in the docking port frame
$\boldsymbol{R}^o_{\text{dock}}$ = position docking port in the object body frame
$\boldsymbol{R}^o_{\text{featr}}$ = position object feature in the object body frame
$\boldsymbol{R}^c_{\text{optrk}}$ = position of the optical tracking camera in the chaser body frame
$\boldsymbol{R}^{\text{lvlh}}$ = position of chaser relative an object centered rotating local-vertical/local-horizontal (lvlh) frame
$\boldsymbol{r}^i_c$ = position of chaser in an inertial coordinate frame
$\boldsymbol{r}^i_o$ = position of object in an inertial coordinate frame
$\mathcal{T}^a_b$ = direction cosine matrix representing the orientation of the b frame with respect to the a frame
$\boldsymbol{T}^b_{\text{aero}}$ = aerodynamic torque in body a coordinate frame
$\boldsymbol{T}^b_{\text{grav}}$ = gravity-gradient torque in a body coordinate frame
$\boldsymbol{T}^b_{\text{wheel}}$ = momentum wheel control torque in a body coordinate frame
$\mathcal{T}^i_c$ = direction cosine matrix representation of $\boldsymbol{q}^i_c$
$\mathcal{T}^i_o$ = direction cosine matrix representation of $\boldsymbol{q}^i_o$
$\hat{\boldsymbol{u}}$ = continuous actuator commands issued by the flight computer
$\boldsymbol{V}^d_{\text{des}}$ = desired velocity of chaser attach point relative to the object docking port in the docking port frame
$\boldsymbol{V}^{\text{lvlh}}$ = velocity of chaser relative an object centered rotating lvlh frame
$\boldsymbol{v}$ = vector indicated by boldface type
$\boldsymbol{v}^a$ = vector represented in coordinate frame a
$\boldsymbol{v}^i_c$ = velocity of chaser in an inertial coordinate frame
$\boldsymbol{v}^i_o$ = velocity of object in an inertial coordinate frame
$\boldsymbol{x}$ = true states
$\hat{\boldsymbol{x}}$ = flight computer navigation states
$\bar{\boldsymbol{x}}$ = reference states
$\tilde{\boldsymbol{y}}$ = continuous inertial measurements
$\tilde{\boldsymbol{z}}_k$ = discrete noninertial measurements at time t_k
$\Delta\hat{\boldsymbol{u}}_j$ = impulsive actuator commands issued by the flight computer at time t_j
$\Delta\boldsymbol{v}^i_{\text{act}}$ = actuator Δv in an inertial coordinate frame
$\Delta\tilde{\boldsymbol{y}}_j$ = discrete inertial measurements at time t_j
$\delta_{jj'}$ = Kronecker delta function
$\delta\boldsymbol{e}$ = true navigation error
$\delta\boldsymbol{q}(\boldsymbol{\epsilon})$ = quaternion associated with a small rotation,
$$\delta\boldsymbol{q}(\boldsymbol{\epsilon}) \approx \begin{pmatrix} \boldsymbol{\epsilon}/2 \\ 1 \end{pmatrix}$$
$\delta\mathcal{T}(\boldsymbol{\epsilon})$ = direction cosine matrix associated with a small rotation, $\delta\mathcal{T}(\boldsymbol{\epsilon}) \approx I - [\boldsymbol{\epsilon}\times]$
$\delta(t-t')$ = Dirac delta function
$\delta\boldsymbol{x}$ = true state dispersions
$\delta\hat{\boldsymbol{x}}$ = navigation state dispersions
$\boldsymbol{\epsilon}$ = small rotation vector
$[\boldsymbol{\epsilon}\times]$ = cross-product matrix defined by the ordinary cross product $[\boldsymbol{\epsilon}\times]\boldsymbol{v} = \boldsymbol{\epsilon}\times\boldsymbol{v}$

Received 11 August 2005; presented as Paper 2005–5856 at the AIAA Guidance, Navigation, and Control Conference, San Francisco, CA, 15–18 August 2005, revision received 12 December 2005; accepted for publication 13 December 2005. Copyright © 2006 by David K. Geller. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0731-5090/06 $10.00 in correspondence with the CCC.

*Assistant Professor, Mechanical and Aerospace Engineering Department. Senior Member AIAA.

$\boldsymbol{\omega}_c^c$ = angular velocity of the chaser with respect to the inertial frame represented in the chaser frame

$\boldsymbol{\omega}_o^o$ = angular velocity of the object with respect to the inertial frame represented in the object frame

$\otimes$ = quaternion multiplication operator such that $\boldsymbol{q}_c^a = \boldsymbol{q}_c^b \otimes \boldsymbol{q}_b^a$ corresponds to the sequence of rotations $\mathcal{T}_c^a = \mathcal{T}_c^b \mathcal{T}_b^a$

Superscripts

$\hat{}$ = parameters, variables, and functions associated with the flight algorithms

$\sim$ = measured values

$\bar{}$ = reference values

I. Introduction

THE objective of this paper is threefold: 1) formulate and develop linear covariance analysis equations that can be applied to many different types of guidance, navigation, and control (GN&C) problems, 2) develop the models and equations for a six-degree-of-freedom linear covariance rendezvous analysis problem, and 3) demonstrate that linear covariance techniques can be used for rendezvous analysis and have the potential to be used for autonomous onboard mission planning.

A general linear covariance program for GN&C analysis must include the effects of environment, actuator and sensor uncertainties, estimation errors, and most importantly, the effect of the uncertainties and estimations errors on trajectory and attitude control errors, which are known as dispersions from the desired nominal path. Covariance analysis is commonly used to study the effects of environment and actuator uncertainties on trajectory dispersions.[1–6] Covariance analysis is also commonly used to assess estimation errors and the effect of modeling errors on the estimation errors.[7–9] This is sometimes called consider analysis.[10,11] However, to properly assess the performance of a closed-loop GN&C system, the combined effect of estimation errors and environmental, sensor, and actuator uncertainties on trajectory and attitude dispersions must be studied. Battin[12] develops a linear covariance methodology in which position and velocity trajectory dispersions are coupled to position and velocity estimation errors through the calculation of onboard impulsive maneuvers (i.e., velocity state corrections) based on the estimated navigation state. This formulation is quite useful because it quantifies the effects of navigation estimation errors on trajectory dispersions. It does not capture the effect of modeling errors on the estimation error, and it does not account for the effect of inertial measurement errors (gyro and accelerometer) on trajectory and attitude dispersions. Maybeck[13] provides a very general linear covariance formulation in which the covariances of true state dispersions and true filter state estimation errors are determined, as well as the flight computer's estimate of the filter state covariance. This formulation allows for discrete state corrections, but it does not account for the effects of sensor error on state propagation (i.e., inertial measurement errors), and it does not account for the effects of continuous state corrections or continuous control feedback on state dispersions. The linear covariance program developed in this paper is presented in a generalized format that will encompass both Battin's formulation and Maybeck's formulation, as well as several additional features including continuous state corrections and inertial sensors measurements.

With the recent deployment and potential development of several new rendezvous missions[14,15] such as XSS-11, DART (demonstration of autonomous rendezvous techniques), Orbital Express, and the Hubble Robotic Servicing and Deorbit Mission, the need for a six-degree-of-freedom closed-loop rendezvous GN&C performance analysis capability is increased. Monte Carlo programs and other deterministic rendezvous programs often provide this capability, but a deterministic approach can be time consuming. The covariance analysis program developed in this paper provides an alternative and complementary analysis program that requires less time to execute and fills what is currently an apparent void in the area of rendezvous GN&C covariance analysis.

In Sec. II a general methodology for developing GN&C linear covariance programs from Monte Carlo programs is presented. Section III describes the truth models for a Monte Carlo rendezvous simulation, and Sec. IV describes the rendezvous flight algorithms. These models and algorithms are used to develop a LinCov program that is applied in Sec. V to two simple problems: a rendezvous mission planning problem and a rendezvous analysis problem. Concluding remarks are provided in Sec. VI.

II. Methods

Linear covariance (LinCov) techniques are designed to produce the same statistical results as a Monte Carlo simulation without doing hundreds or thousands of simulations. There is initial overhead associated with the development of linear models, but these costs are offset by having the capability to study many different aspects of a GN&C problem in a short period of time.

Figure 1 illustrates a generic GN&C Monte Carlo program. The variables in this figure will be defined shortly, but a brief summary is appropriate. White-noise processes $\boldsymbol{w}$, $\Delta\boldsymbol{w}_j$, $\boldsymbol{\eta}$, $\Delta\boldsymbol{\eta}_j$, $\boldsymbol{\nu}_k$, and flight computer actuator commands $\hat{\boldsymbol{u}}$, $\Delta\hat{\boldsymbol{u}}_j$, drive truth models, which in turn generate the true state $\boldsymbol{x}$ of the system along with simulated sensor measurements $\tilde{\boldsymbol{z}}_k$, $\tilde{\boldsymbol{y}}$, $\Delta\tilde{\boldsymbol{y}}_j$. A navigation algorithm processes the sensor data and produces a navigation state $\hat{\boldsymbol{x}}$ and the navigation state error covariance $\hat{P}$. The navigation state is used by the targeting, pointing, and control functions of the flight computer to generate the actuator commands. The key variables are the true dispersions from the reference $\delta\boldsymbol{x} = \boldsymbol{x} - \bar{\boldsymbol{x}}$, the navigation dispersions from the reference $\delta\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}$, and the true navigation errors $\delta\boldsymbol{e} = \delta\hat{\boldsymbol{x}} - M_x\delta\boldsymbol{x}$.

In a Monte Carlo program, the covariance of the dispersions and navigation errors is determined by collecting and compiling the results of N simulations:

$$D_{\text{true}} \approx \frac{1}{N-1}\sum_{i=1}^{N}\delta\boldsymbol{x}\delta\boldsymbol{x}^T, \qquad D_{\text{nav}} \approx \frac{1}{N-1}\sum_{i=1}^{N}\delta\hat{\boldsymbol{x}}\delta\hat{\boldsymbol{x}}^T$$

$$P_{\text{true}} \approx \frac{1}{N-1}\sum_{i=1}^{N}\delta\boldsymbol{e}\delta\boldsymbol{e}^T \tag{1}$$

The covariance of trajectory and attitude dispersions, navigation errors, and Δv or fuel usage is often of primary interest. Navigation performance is determined by evaluating and comparing P_{true} and $\hat{P}$.

In a linear covariance approach, the covariances in Eq. (1) are generated in a single simulation by directly propagating, updating, and correcting an augmented state covariance matrix $\boldsymbol{C}_A$ that will be defined shortly. The LinCov program is based on a linearized version of the Monte Carlo program and will produce Monte Carlo like results in a fraction of the time.

There are limitations and caveats associated with LinCov programs primarily caused by the effects of nonlinearities. However, it is often the case that the expected operational range of a system falls within a narrow band that can be accurately described by linear equations. This is especially true for orbital dynamics where the expected envelope of trajectories about the nominal is often very small. Some effects however (e.g., quantization error, deadbands,

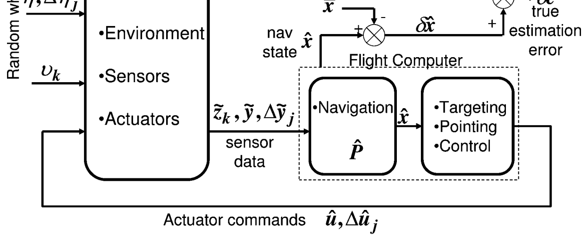

Fig. 1 Generic Monte Carlo simulation for GN&C analysis.

and hysteresis) cannot be linearized, and so the LinCov approach designer must select other representation for these effects such as statistical linearization[6] or simply a Gaussian distribution with a conservative variance. Using a combination of analytic methods, numerical methods, and engineering judgment, issues associated with highly nonlinear phenomena can often be circumvented, and an accurate linear model of the closed-loop GN&C system can be developed.

A. Nonlinear Modeling

The dynamics of the Monte Carlo truth models in Fig. 1 are represented by

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \hat{\boldsymbol{u}}, t) + \boldsymbol{w} \tag{2}$$

where $\boldsymbol{x}$ is a vector of n true states, $\hat{\boldsymbol{u}}$ is a vector of $n_{\hat{u}}$ actuator commands issued by the flight computer, and $\boldsymbol{w}$ is a vector of zero-mean white-noise processes with covariance

$$E[\boldsymbol{w}(t)\boldsymbol{w}^T(t')] = S_w(t)\delta(t - t') \tag{3}$$

A detailed description of the true state vector and its associated dynamics for the rendezvous problem is given in Sec. III.

Sensors are divided into two distinct categories: inertial sensors used for navigation state propagation and noninertial sensors for navigation state updates. The inertial instruments (e.g., gyros and accelerometers) are modeled by nonlinear equations and provide a vector of n_y continuous measurements $\tilde{\boldsymbol{y}}$ and a vector of $n_{\Delta y}$ discrete measurements $\Delta\tilde{\boldsymbol{y}}_j$ at time t_j, where

$$\tilde{\boldsymbol{y}} = \boldsymbol{c}(\boldsymbol{x}, t) + \boldsymbol{\eta} \tag{4}$$

$$\Delta\tilde{\boldsymbol{y}}_j = \Delta\boldsymbol{c}(\boldsymbol{x}_j, t_j) + \Delta\boldsymbol{\eta}_j \tag{5}$$

The discrete inertial measurements provide a model for Δv measurements when maneuvers are infrequent and relatively short. Noninertial instruments (e.g., cameras, lidar, radar) provide n_z discrete measurements $\tilde{z}_k$ at times t_k, where

$$\tilde{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, t_k) + \boldsymbol{\nu}_k \tag{6}$$

The covariance of the measurement noise is given by

$$E[\boldsymbol{\eta}(t)\boldsymbol{\eta}^T(t')] = S_\eta(t)\delta(t - t') \tag{7}$$

$$E\left[\Delta\boldsymbol{\eta}_j\Delta\boldsymbol{\eta}_{j'}^T\right] = S_{\Delta\eta}(t_j)\delta_{jj'} \tag{8}$$

$$E\left[\boldsymbol{\nu}_k\boldsymbol{\nu}_{k'}^T\right] = R_\nu(t_k)\delta_{kk'} \tag{9}$$

Instantaneous corrections to the state vector such as impulsive translation or rotational maneuvers are also allowed. Corrections at time t_j are represented as

$$\boldsymbol{x}_j^{+c} = \boldsymbol{x}_j^{-c} + \boldsymbol{d}\left(\boldsymbol{x}_j^{-c}, \Delta\hat{\boldsymbol{u}}_j, t_j\right) + \Delta\boldsymbol{w}_j \tag{10}$$

where the correction $\boldsymbol{d}(\boldsymbol{x}_j^{-c}, \Delta\hat{\boldsymbol{u}}_j, t_j)$ is a function of $\boldsymbol{x}_j^{-c}$, the true state just before the correction is applied, and $\Delta\hat{\boldsymbol{u}}_j$, a vector of $n_{\Delta\hat{u}}$ instantaneous discrete corrective actuator commands issued by the flight computer. A random error $\Delta\boldsymbol{w}_j$ with covariance

$$E\left[\Delta\boldsymbol{w}_j\Delta\boldsymbol{w}_{j'}^T\right] = S_{\Delta w}(t_j)\delta_{jj'} \tag{11}$$

is also applied during correction.

Thus, the Monte Carlo truth models are defined by the continuous-time functions $\boldsymbol{f}(\boldsymbol{x}, \hat{\boldsymbol{u}}, t)$ and $\boldsymbol{c}(\boldsymbol{x}, t)$, the discrete-time functions $\Delta\boldsymbol{c}(\boldsymbol{x}_j, t_j)$, $\boldsymbol{h}(\boldsymbol{x}_k, t_k)$, and $\boldsymbol{d}(\boldsymbol{x}_j^{-c}, \Delta\hat{\boldsymbol{u}}_j, t_j)$, and the covariance matrices $S_w(t)$, $S_{\Delta w}(t_j)$, $S_\eta(t)$, $S_{\Delta\eta}(t_j)$, and $R_\nu(t_k)$.

Next, the flight algorithms that produce the actuator commands $\hat{\boldsymbol{u}}$ and $\Delta\hat{\boldsymbol{u}}_j$ in Eqs. (2) and (10) must be defined. This includes navigation, control, pointing, and targeting. Remember that a caret refers to variables and functions related to flight algorithms, not estimates or expected values. Examples are actuator commands $\hat{\boldsymbol{u}}$, flight computer variables such as $\hat{\boldsymbol{x}}$ or $\hat{z}$, and function evaluations $\hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}, t)$.

The flight algorithms are divided into four sequential processes: 1) navigation state and state covariance propagation; 2) navigation state and state covariance update; 3) navigation state and state covariance correction; and 4) pointing, maneuver targeting, and translation and rotational control.

The navigation state and state covariance propagation algorithms are written as

$$\dot{\hat{\boldsymbol{x}}} = \hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}, \tilde{\boldsymbol{y}}, t) \tag{12}$$

$$\dot{\hat{P}} = [\hat{F}_{\hat{x}} + \hat{F}_{\tilde{y}}\hat{C}_{\hat{x}}]\hat{P} + \hat{P}[\hat{F}_{\hat{x}} + \hat{F}_{\tilde{y}}\hat{C}_{\hat{x}}]^T + \hat{F}_{\tilde{y}}\hat{S}_\eta\hat{F}_{\tilde{y}}^T + \hat{S}_w \tag{13}$$

where $\hat{\boldsymbol{x}}$ is the navigation state of dimension $\hat{n} < n$, and the true values of the navigation state $\boldsymbol{x}_n$ are given by a mapping function of the true states.

$$\boldsymbol{x}_n = m(\boldsymbol{x}) \tag{14}$$

$\hat{P}$ represents the flight computer's assessment of the navigation state error covariance matrix, and $\hat{F}_{\hat{x}}$, $\hat{F}_{\tilde{y}}$, and $\hat{C}_{\hat{x}}$ are the partial derivatives of $\hat{\boldsymbol{f}}$ and $\hat{\boldsymbol{c}}$ with respect to the variable indicated by the subscript. The partial derivatives can be evaluated at $\hat{\boldsymbol{x}}$ as they would be for an extended Kalman filter, or they can be evaluated along a reference trajectory $\bar{\boldsymbol{x}}$ and stored in memory for a standard Kalman filter implementation. Additional variance as a result of inertial instrument measurement noise is accounted for by $\hat{S}_\eta$, and the variance as a result of unmodeled accelerations is represented by $\hat{S}_w$.

The navigation state and state covariance update algorithms[11] can be written as

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \hat{K}(t_k)[\hat{z}_k - \hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}_k, t_k)] \tag{15}$$

$$\hat{P}\left(t_k^+\right) = [I - \hat{K}(t_k)\hat{H}_{\hat{x}}(t_k)]\hat{P}\left(t_k^-\right)[I - \hat{K}(t_k)\hat{H}_{\hat{x}}(t_k)]^T + \hat{K}(t_k)\hat{R}_\nu(t_k)\hat{K}^T(t_k) \tag{16}$$

where the Joseph formulation[13] is used for the covariance update and the Kalman gain $\hat{K}(t_k)$ is given by

$$\hat{K}(t_k) = \hat{P}(t_k)\hat{H}_{\hat{x}}^T(t_k)\left[\hat{H}_{\hat{x}}(t_k)\hat{P}(t_k)\hat{H}_{\hat{x}}^T(t_k) + \hat{R}_\nu(t_k)\right]^{-1} \tag{17}$$

The measurement sensitivity matrix is given by $\hat{H}_{\hat{x}}$, and the measurement noise covariance is $\hat{R}_\nu$.

Next, when an instantaneous state correction as in Eq. (10) occurs, the navigation state also must be corrected. The correction algorithm can be written as

$$\hat{\boldsymbol{x}}_j^{+c} = \hat{\boldsymbol{x}}_j^{-c} + \hat{\boldsymbol{d}}\left(\hat{\boldsymbol{x}}^{-c}, \Delta\hat{\boldsymbol{u}}_j, \Delta\tilde{\boldsymbol{y}}_j, t_j\right) \tag{18}$$

$$\hat{P}\left(t_j^{+c}\right) = [I + \hat{D}_{\hat{x}}(t_j) + \hat{D}_{\Delta\tilde{y}}(t_j)\Delta\hat{C}_{\hat{x}}(t_j)]\hat{P}^{-c}(t_j)[I + \hat{D}_{\hat{x}}(t_j) + \hat{D}_{\Delta\tilde{y}}(t_j)\Delta\hat{C}_{\hat{x}}(t_j)]^T + \hat{D}_{\Delta\tilde{y}}(t_j)\hat{S}_{\Delta\eta}\hat{D}_{\Delta\tilde{y}}(t_j)^T + \hat{S}_{\Delta w}(t_j) \tag{19}$$

where $\hat{D}_{\hat{x}}$, $\hat{D}_{\Delta\tilde{y}}$, and $\Delta\hat{C}_{\hat{x}}$ are the partial derivatives of $\hat{\boldsymbol{d}}$ and $\Delta\hat{\boldsymbol{c}}$ with respect to the variable indicated by the subscript. For an instantaneous correction, variance caused by inertial measurement noise is represented by $\hat{S}_{\Delta\eta}$ and variance caused by the noise component of the correction application error is represented by $\hat{S}_{\Delta w}$. For example, if the correction is an impulsive maneuver,$\hat{S}_{\Delta\eta}$ corresponds to discrete Δv measurement noise, and $\hat{S}_{\Delta w}$ corresponds to the noise component of the Δv application error.

The pointing, targeting, and control algorithms (see Fig. 1) use the most recent value of the navigation state to compute continuous and discrete actuator commands

$$\hat{\boldsymbol{u}} = \hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}, t) \tag{20}$$

$$\Delta\hat{\boldsymbol{u}}_j = \Delta\hat{\boldsymbol{g}}\left(\hat{\boldsymbol{x}}_j^{-c}, t_j\right) \tag{21}$$

where $\hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}, t)$ represents the algorithms for continuous commanding and $\Delta\hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}_j^{-c}, t_j)$ represents the algorithms for discrete instantaneous commanding.

Thus the flight algorithms are completely defined by the continuous-time functions $\hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}, \tilde{\boldsymbol{y}}, t)$, $\hat{\boldsymbol{c}}(\hat{\boldsymbol{x}}, t)$ and $\hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}, t)$; the discrete-time functions $\Delta\hat{\boldsymbol{c}}(\hat{\boldsymbol{x}}_j, t_j)$, $\Delta\hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}_j^{-c}, t_j)$, $\hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}_k, t_k)$; and $\hat{\boldsymbol{d}}(\hat{\boldsymbol{x}}^{-c}, \Delta\hat{\boldsymbol{u}}_j, \Delta\tilde{\boldsymbol{y}}_j, t_j)$; and the covariance matrices $\hat{S}_w(t)$, $\hat{S}_{\Delta w}(t_j)$, $\hat{S}_\eta(t)$, $\hat{S}_{\Delta\eta}(t_j)$, and $\hat{R}_\nu(t_k)$.

Note that the preceding navigation algorithm has been written in a form such that it can propagate the navigation state using data from inertial instruments $\tilde{\boldsymbol{y}}$, $\Delta\tilde{\boldsymbol{y}}_j$, or data from empirical models of the actuators. A combination of both propagation techniques (e.g., gyro measurements to propagate rotational dynamics and empirical thruster models to propagate translation dynamics) is also possible.

B. Linear Modeling

The preceding equations are linearized about a reference trajectory defined by $\bar{\boldsymbol{x}}(t)$ to produce a set of equations that describe the time evolution of the true state dispersions from the reference, $\delta\boldsymbol{x}(t) = \boldsymbol{x}(t) - \bar{\boldsymbol{x}}(t)$, and the time evolution of the navigation state dispersions from the reference, $\delta\hat{\boldsymbol{x}}(t) = \hat{\boldsymbol{x}}(t) - \bar{\boldsymbol{x}}(t)$, as shown in Fig. 1.

The propagation equations (2) and (12) and (4) and (20) are linearized to produce

$$\delta\dot{\boldsymbol{x}} = F_x\delta\boldsymbol{x} + F_{\hat{u}}\hat{G}_{\hat{x}}\delta\hat{\boldsymbol{x}} + \boldsymbol{w} \tag{22}$$

$$\delta\dot{\hat{\boldsymbol{x}}} = (\hat{F}_{\hat{x}} + \hat{F}_{\hat{u}}\hat{G}_{\hat{x}})\delta\hat{\boldsymbol{x}} + \hat{F}_{\tilde{y}}C_x\delta\boldsymbol{x} + \hat{F}_{\tilde{y}}\boldsymbol{\eta} \tag{23}$$

where uppercase characters denote partial derivatives taken with respect the variable indicated by subscript and evaluated along the reference trajectory (e.g., $F_x = \partial\boldsymbol{f}/\partial\boldsymbol{x}|_{\bar{x}}$, $\hat{G}_{\hat{x}} = \partial\hat{\boldsymbol{g}}/\partial\hat{\boldsymbol{x}}|_{\bar{x}}$, $F_{\hat{u}} = \partial\boldsymbol{f}/\partial\hat{\boldsymbol{u}}|_{\bar{x}}$, etc).

The state update given by Eq. (15) and the measurement equation in Eq. (6) are also linearized about the reference trajectory to produce

$$\delta\boldsymbol{x}_k^+ = \delta\boldsymbol{x}_k^- \tag{24}$$

$$\delta\hat{\boldsymbol{x}}_k^+ = \hat{K}(t_k)H_x(t_k)\delta\boldsymbol{x}_k^- + [I - \hat{K}(t_k)\hat{H}_{\hat{x}}(t_k)]\delta\hat{\boldsymbol{x}}_k^- + \hat{K}(t_k)\boldsymbol{\nu}_k \tag{25}$$

where the dispersions $\delta\boldsymbol{x}$ are unaffected by measurement updates.

Finally, the state corrections in Eqs.(10) and (18) are linearized to produce

$$\delta\boldsymbol{x}_j^{+c} = [I + D_x(t_j)]\delta\boldsymbol{x}_j^{-c} + D_{\Delta\hat{u}}(t_j)\Delta\hat{G}_{\hat{x}}(t_j)\delta\hat{\boldsymbol{x}}_j^{-c} + \Delta\boldsymbol{w}_j \tag{26}$$

$$\delta\hat{\boldsymbol{x}}_j^{+c} = [I + \hat{D}_{\hat{x}}(t_j) + \hat{D}_{\Delta\hat{u}}(t_j)\Delta\hat{G}_{\hat{x}}(t_j)]\delta\hat{\boldsymbol{x}}_j^{-c}$$

$$+ \hat{D}_{\Delta\tilde{y}}(t_j)\Delta C_x(t_j)\delta\boldsymbol{x}_j^{-c} \tag{27}$$

$$+ \hat{D}_{\Delta\tilde{y}}(t_j)\Delta\boldsymbol{\eta}_j \tag{28}$$

where again the partial derivatives are evaluated along the reference trajectory.

It is important to recognize that Eqs. (22–27) clearly show that the navigation dispersions and the true dispersions are strongly coupled through sensor measurements and actuator commands.

C. Augmented Linear System

Next we define an augmented state vector[13] $\boldsymbol{X}$ consisting of the true state dispersions and navigation state dispersions:

$$\boldsymbol{X} = \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\hat{\boldsymbol{x}} \end{bmatrix} \tag{29}$$

Equations (22–27) can now be written in the following compact form:

$$\dot{\boldsymbol{X}} = \mathcal{F}\boldsymbol{X} + \mathcal{G}\boldsymbol{\eta} + \mathcal{W}\boldsymbol{w} \tag{30}$$

$$\boldsymbol{X}_k^+ = \mathcal{A}_k\boldsymbol{X}_k^- + \mathcal{B}_k\boldsymbol{\nu}_k \tag{31}$$

$$\boldsymbol{X}_j^{+c} = \mathcal{D}_j\boldsymbol{X}_j^{-c} + \mathcal{M}_j\Delta\boldsymbol{\eta}_j + \mathcal{N}_j\Delta\boldsymbol{w}_j \tag{32}$$

where

$$\mathcal{F} = \begin{pmatrix} F_x & F_{\hat{u}}\hat{G}_{\hat{x}} \\ \hat{F}_{\tilde{y}}C_x & \hat{F}_{\hat{x}} + \hat{F}_{\hat{u}}\hat{G}_{\hat{x}} \end{pmatrix}, \quad \mathcal{G} = \begin{pmatrix} 0_{n\times n_y} \\ \hat{F}_{\tilde{y}} \end{pmatrix}, \quad \mathcal{W} = \begin{pmatrix} I_{n\times n} \\ 0_{\hat{n}\times n} \end{pmatrix} \tag{33}$$

$$\mathcal{A}_k = \begin{pmatrix} I_{n\times n} & 0_{n\times\hat{n}} \\ \hat{K}(t_k)H_x(t_k) & I_{\hat{n}\times\hat{n}} - \hat{K}(t_k)\hat{H}_{\hat{x}} \end{pmatrix}, \qquad \mathcal{B}_k = \begin{pmatrix} 0_{n\times n_z} \\ \hat{K}(t_k) \end{pmatrix} \tag{34}$$

$$\mathcal{D}_j = \begin{pmatrix} I_{n\times n} + D_x(t_j) & D_{\Delta\hat{u}}(t_j)\Delta\hat{G}_{\hat{x}}(t_j) \\ \hat{D}_{\Delta\tilde{y}}(t_j)\Delta C_x(t_j) & I_{\hat{n}\times\hat{n}} + \hat{D}_{\hat{x}}(t_j) + \hat{D}_{\Delta\hat{u}}(t_j)\Delta\hat{G}_{\hat{x}}(t_j) \end{pmatrix} \tag{35}$$

$$\mathcal{M}_j = \begin{pmatrix} 0_{n\times n_{\Delta y}} \\ \hat{D}_{\Delta\tilde{y}}(t_j) \end{pmatrix}, \qquad \mathcal{N}_j = \begin{pmatrix} I_{n\times n} \\ 0_{\hat{n}\times n} \end{pmatrix}$$

D. Covariance Equations

We can now formulate the covariance equations for the entire Monte Carlo simulation. Remember that with respect to the Monte Carlo simulation, the true state $\boldsymbol{x}(t)$ and the navigated state $\hat{\boldsymbol{x}}(t)$ are random processes that vary from one Monte Carlo sample to the next! Because $E[\boldsymbol{x}(t)] = E[\bar{\boldsymbol{x}}(t)]$ and $E[\hat{\boldsymbol{x}}(t)] = E[\boldsymbol{x}(t)]$, the true and navigation state dispersions $\delta\boldsymbol{x}(t)$ and $\delta\hat{\boldsymbol{x}}(t)$ will have zero means, and thus $E[\boldsymbol{X}(t)] = 0$. The covariance of the augmented state $\boldsymbol{X}(t)$, $\boldsymbol{C}_A = E[\boldsymbol{X}(t)\boldsymbol{X}^T(t)]$ is obtained from the following set of covariance propagation, update, and correction equations:

$$\dot{\boldsymbol{C}}_A = \mathcal{F}\boldsymbol{C}_A + \boldsymbol{C}_A\mathcal{F}^T + \mathcal{G}S_\eta\mathcal{G}^T + \mathcal{W}S_w\mathcal{W}^T \tag{36}$$

$$\boldsymbol{C}_A\left(t_k^+\right) = \mathcal{A}_k\boldsymbol{C}_A\left(t_k^-\right)\mathcal{A}_k^T + \mathcal{B}_kR_\nu(t_k)\mathcal{B}_k^T \tag{37}$$

$$\boldsymbol{C}_A\left(t_j^{+c}\right) = \mathcal{D}_j\boldsymbol{C}_A\left(t_j^{-c}\right)\mathcal{D}_j^T + \mathcal{M}_jS_{\Delta\eta}(t_j)\mathcal{M}_j^T + \mathcal{N}_jS_{\Delta w}(t_j)\mathcal{N}_j^T \tag{38}$$

where it is assumed that $\boldsymbol{w}$, $\Delta\boldsymbol{w}_j$, $\boldsymbol{\eta}$, $\Delta\boldsymbol{\eta}_j$, and $\boldsymbol{\nu}_k$ are mutually uncorrelated.

When the onboard flight computer covariance equations (13), (16), (17), and (19) are appended to Eqs. (36–38), a complete set of linear covariance analysis equations is formed.

E. Performance Evaluation

The overall closed-loop performance of a GN&C system is evaluated by examining the covariance of the true state dispersions D_{true}, the covariance of the navigation state dispersions D_{nav}, and the covariance of the true navigation state errors P_{true}. In a Monte Carlo program (see Fig. 1), these covariances are computed according to Eq. (1). In the LinCov program they are computed as follows:

$$D_{\text{true}} = E[\delta\boldsymbol{x}(t)\delta\boldsymbol{x}^T(t)] = (I_{n\times n} \quad 0_{n\times\hat{n}})\boldsymbol{C}_A(I_{n\times n} \quad 0_{n\times\hat{n}})^T \tag{39}$$

$$D_{\text{nav}} = E[\delta\hat{\boldsymbol{x}}(t)\delta\hat{\boldsymbol{x}}^T(t)] = (0_{\hat{n}\times n} \quad I_{\hat{n}\times\hat{n}})\boldsymbol{C}_A(0_{\hat{n}\times n} \quad I_{\hat{n}\times\hat{n}})^T \tag{40}$$

$$P_{\text{true}} = E\left[\{\delta\hat{\boldsymbol{x}}(t) - M_x\delta\boldsymbol{x}(t)\}\{\delta\hat{\boldsymbol{x}}(t) - M_x\delta\boldsymbol{x}(t)\}^T\right] \tag{41}$$

$$= (-M_x \quad I_{\hat{n}\times\hat{n}})\boldsymbol{C}_A(-M_x \quad I_{\hat{n}\times\hat{n}})^T \tag{42}$$

where M_x is the partial with respect to the true state of the mapping function defined in Eq. (14).

The true navigation errors P_{true} can then be compared to $\hat{P}$ to evaluate the performance of the onboard navigation system, and the performance of the closed-loop trajectory control system is determined by examining D_{true}.

III. Monte Carlo Truth Models for Rendezvous

The truth models in Fig. 1 generally include environment force and torque models, translation and rotational dynamics models, sensor measurement models, and actuator force and torque models. The models populate the linear covariance equations developed in the preceding section

For this Monte Carlo program it is assumed that the chaser spacecraft has strap-down gyros and a star camera for attitude determination and an optical tracking camera for relative navigation. Attitude control is accomplished with momentum wheels, and translation control is accomplished with a system of thrusters. The target vehicle, also referred to as a resident space object, is assumed to be in a passive mode with no active translation or attitude control. It is assumed that the chaser is in the vicinity of the target (range < 20 km), and only small brief maneuvers that can be modeled impulsively are required.

The truth state of the Monte Carlo program is an n-dimensional vector defined by

$$\boldsymbol{x} = (\boldsymbol{x}_o, \boldsymbol{x}_c, \boldsymbol{p})^T \tag{43}$$

It consists of 13 resident space object states $\boldsymbol{x}_o$, 13 chaser states $\boldsymbol{x}_c$, and 57 parameter states $\boldsymbol{p}$. The state of the resident space object is composed of the object's inertial position, velocity, attitude quaternion, and angular velocity:

$$\boldsymbol{x}_o = \left(\boldsymbol{r}_o^i, \boldsymbol{v}_o^i, \boldsymbol{q}_o^i, \boldsymbol{\omega}_o^o\right)^T \tag{44}$$

The state of the chaser consists of the chaser's inertial position, velocity, attitude quaternion, and angular velocity:

$$\boldsymbol{x}_c = \left(\boldsymbol{r}_c^i, \boldsymbol{v}_c^i, \boldsymbol{q}_c^i, \boldsymbol{\omega}_c^c\right)^T \tag{45}$$

The n_p-dimensional parameter state consists of sensor, actuator, and environment parameters

$$\boldsymbol{p} = \left(\boldsymbol{p}_{\text{gyro}}, \boldsymbol{p}_{\text{starcam}}, \boldsymbol{p}_{\text{optrk}}, \boldsymbol{p}_{\text{torque}}, \boldsymbol{p}_{\Delta v}, \boldsymbol{p}_{\text{atm}}, \boldsymbol{p}_{\text{aero}_o}, \boldsymbol{p}_{\text{aero}_c}\right)^T \tag{46}$$

that are typically constant biases or time-varying biases with known time constants and modeled as first-order Markov processes. The first-order Markov models are often sufficient for GN&C analysis, but higher-order models can be accommodated if needed. These parameters will be defined in the following discussion.

A. Dynamics

The dynamics for the true state vector are given by

$$\dot{\boldsymbol{r}}_o^i = \boldsymbol{v}_o^i \tag{47}$$

$$\dot{\boldsymbol{v}}_o^i = \left[\boldsymbol{F}^i_{\text{aero}_o}\left(\boldsymbol{r}_o^i, \boldsymbol{v}_o^i, \boldsymbol{p}_{\text{aero}_o}, \boldsymbol{p}_{\text{atm}}\right) + \boldsymbol{F}^i_{\text{grav}_o}\left(\boldsymbol{r}_o^i\right)\right]\Big/ m_o \tag{48}$$

$$\dot{\boldsymbol{q}}_o^i = \tfrac{1}{2}\boldsymbol{\omega}_o^o \otimes \boldsymbol{q}_o^i \tag{49}$$

$$\dot{\boldsymbol{\omega}}_o^o = I_o^{-1}\Big[\boldsymbol{T}^o_{\text{aero}_o}\left(\boldsymbol{r}_o^i, \boldsymbol{v}_o^i, \boldsymbol{q}_o^i, \boldsymbol{p}_{\text{aero}_o}, \boldsymbol{p}_{\text{atm}}\right) + \boldsymbol{T}^o_{\text{grav}_o}\left(\boldsymbol{r}_o^i, \boldsymbol{q}_o^i\right) - \boldsymbol{\omega}_o^o \times I_o \boldsymbol{\omega}_o^o\Big] \tag{50}$$

$$\dot{\boldsymbol{r}}_c^i = \boldsymbol{v}_c^i \tag{51}$$

$$\dot{\boldsymbol{v}}_c^i = \left[\boldsymbol{F}^i_{\text{aero}_c}\left(\boldsymbol{r}_c^i, \boldsymbol{v}_c^i, \boldsymbol{p}_{\text{aero}_c}, \boldsymbol{p}_{\text{atm}}\right) + \boldsymbol{F}^i_{\text{grav}_c}\left(\boldsymbol{r}_c^i\right)\right]\Big/ m_c \tag{52}$$

$$\dot{\boldsymbol{q}}_c^i = \tfrac{1}{2}\boldsymbol{\omega}_c^c \otimes \boldsymbol{q}_c^i \tag{53}$$

$$\dot{\boldsymbol{\omega}}_c^c = I_c^{-1}\Big[\boldsymbol{T}^c_{\text{wheel}}\left(\hat{\boldsymbol{T}}^c_{\text{com}}, \boldsymbol{p}_{\text{act}}\right) + \boldsymbol{T}^c_{\text{aero}_c}\left(\boldsymbol{r}_c^i, \boldsymbol{v}_c^i, \boldsymbol{q}_c^i, \boldsymbol{p}_{\text{aero}_c}, \boldsymbol{p}_{\text{atm}}\right) + \boldsymbol{T}^c_{\text{grav}_c}(\boldsymbol{r}_c^i, \boldsymbol{q}_c^i) \tag{54}$$

$$- \boldsymbol{\omega}_c^c \times I_c \boldsymbol{\omega}_c^c\Big] \tag{55}$$

$$\dot{p}_i = -p_i/\tau_i + w_{p_i}, \qquad i = 1, 2, 3, \ldots, n_p \tag{56}$$

where m_o, I_o are the resident space object mass and inertia and m_c, I_c are the chaser mass and inertia. $\boldsymbol{T}^c_{\text{wheel}}$ is the torque produced by the momentum wheels, and $\Delta\boldsymbol{v}^i_{\text{act}}$ are impulsive maneuvers used to corrective the chaser velocity:

$$\left(\boldsymbol{v}_c^i\right)^{+c} = \left(\boldsymbol{v}_c^i\right)^{-c} + \Delta\boldsymbol{v}^i_{\text{act}}\left(\Delta\hat{\boldsymbol{v}}^c_{\text{com}}, \boldsymbol{q}_c^i, \boldsymbol{p}_{\Delta v}\right) \tag{57}$$

The models for $\Delta\boldsymbol{v}^i_{\text{act}}$ and $\boldsymbol{T}^c_{\text{wheel}}$ are defined in Eqs. (71) and (70), and the parameters p_i are first-order Markov processes driven by white noise with variance:

$$E\left[w_{p_i}(t)w_{p_i}(t')\right] = \sigma^2_{p_i}\delta(t - t'), \qquad i = 1, 2, 3, \ldots, n_p \tag{58}$$

The gravitational forces on the chaser and the object $\boldsymbol{F}^i_{\text{grav}_c}$ and $\boldsymbol{F}^i_{\text{grav}_o}$ are point mass plus J_2 gravity models[16] with random disturbances $\boldsymbol{w}_{g_o}$ and $\boldsymbol{w}_{g_c}$ to represent higher-order unmodeled accelerations. The covariances of the disturbances are

$$E\left[\boldsymbol{w}_{g_o}(t)\boldsymbol{w}^T_{g_o}(t')\right] = E\left[\boldsymbol{w}_{g_c}(t)\boldsymbol{w}^T_{g_c}(t')\right] = S_{w_{\text{grav}}}\delta(t - t') \tag{59}$$

$$E\left[\boldsymbol{w}_{g_o}(t)\boldsymbol{w}^T_{g_c}(t')\right] = 0 \tag{60}$$

The gravity-gradient torques $\boldsymbol{T}^o_{\text{grav}_o}$ and $\boldsymbol{T}^c_{\text{grav}_c}$ are based on point-mass gravity models.[17]

The aerodynamic forces on the object and chaser $\boldsymbol{F}^i_{\text{aero}_o}$ and $\boldsymbol{F}^i_{\text{aero}_c}$ are based on simple drag models with an exponential atmosphere. The scale height and reference density have nominal values $\bar{h}_s$ and $\bar{\rho}_o$ with uncertainties b_{h_s} and b_{ρ_o}. These uncertainties are the atmospheric parameters, $\boldsymbol{p}_{\text{atm}} = (b_{h_s}, b_{\rho_o})^T$. The drag coefficient, reference area, and the location of the center of pressure for each vehicle have reference values $\bar{C}d$, $\bar{A}$, and $\bar{\boldsymbol{R}}_{\text{cp}}$, and uncertainties b_{cd} for the drag coefficient, b_A for the reference area, and $\boldsymbol{f}_{\text{cp}}$, $\boldsymbol{\epsilon}_{\text{cp}}$, and $\boldsymbol{b}_{\text{cp}}$ for the scale-factor uncertainty, directional error, and magnitude uncertainty in the location of the center of pressure. These uncertainties are the object and the chaser aerodynamic parameters

$$\boldsymbol{p}_{\text{aero}_o} = \left(b_{\text{cd}_o}, b_{A_o}, \boldsymbol{f}_{\text{cp}_o}, \boldsymbol{\epsilon}_{\text{cp}_o}, \boldsymbol{b}_{\text{cp}_o}\right)^T \tag{61}$$

$$\boldsymbol{p}_{\text{aero}_c} = \left(b_{\text{cd}_c}, b_{A_c}, \boldsymbol{f}_{\text{cp}_c}, \boldsymbol{\epsilon}_{cpc}, \boldsymbol{b}_{\text{cp}_c}\right)^T \tag{62}$$

Additional unmodeled aerodynamic accelerations on the object and chaser are represented by $\boldsymbol{w}_{\text{aero}_o}$ and $\boldsymbol{w}_{\text{aero}_c}$, where

$$E\left[\boldsymbol{w}_{\text{aero}_o}(t)\boldsymbol{w}^T_{\text{aero}_o}(t')\right] = S^o_{w_{\text{aero}}}\delta(t - t') \tag{63}$$

$$E\left[\boldsymbol{w}_{\text{aero}_c}(t)\boldsymbol{w}^T_{\text{aero}_c}(t')\right] = S^c_{w_{\text{aero}}}\delta(t - t') \tag{64}$$

$$E\left[\boldsymbol{w}_{\text{aero}_o}(t)\boldsymbol{w}^T_{\text{aero}_c}(t')\right] = 0 \tag{65}$$

The aerodynamic torques on the object are given by

$$\boldsymbol{T}_{\text{aero}_o}\left(\boldsymbol{r}_o^i, \boldsymbol{v}_o^i, \boldsymbol{q}_o^i, \boldsymbol{p}_{\text{aero}_o}, \boldsymbol{p}_{\text{atm}}\right) = \boldsymbol{R}^o_{\text{cp}} \times \mathcal{T}\left(\boldsymbol{q}_o^i\right) \cdot \boldsymbol{F}^i_{\text{aero}_o} \tag{66}$$

where the location of the center of pressure with respect to the center of mass is

$$\boldsymbol{R}^o_{\text{cp}} = \delta\mathcal{T}\left(\epsilon_{\text{cp}_o}\right)\left\{\left[I_{3\times 3} + \text{Diag}\left(\boldsymbol{f}_{\text{cp}_o}\right)\right]\bar{\boldsymbol{R}}^o_{\text{cp}} + b_{\text{cp}_o}\right\} \tag{67}$$

The aerodynamic torques on the chaser are given by similar equations.

These relatively simple models provide great flexibility in modeling aerodynamic uncertainties and are very useful in conducting trajectory control and navigation analysis.

B. Actuator Models

The control torques and maneuver Δv model parameters are scale-factor uncertainty $\boldsymbol{f}$, alignment uncertainty $\boldsymbol{\epsilon}$, and a bias uncertainty $\boldsymbol{b}$:

$$\boldsymbol{p}_{\text{torque}} = (\boldsymbol{f}_{\text{torque}}, \boldsymbol{\epsilon}_{\text{torque}}, \boldsymbol{b}_{\text{torque}})^T \tag{68}$$

$$\boldsymbol{p}_{\Delta v} = (\boldsymbol{f}_{\Delta v}, \boldsymbol{\epsilon}_{\Delta v}, \boldsymbol{b}_{\Delta v})^T \tag{69}$$

In terms of these parameters, the momentum wheel torque in Eq. (54) is given by

$$\begin{aligned}\boldsymbol{T}^c_{\text{wheel}}\left(\hat{\boldsymbol{T}}^c_{\text{com}}, \boldsymbol{p}_{\text{torque}}\right) &= \delta\mathcal{T}(\boldsymbol{\epsilon}_{\text{torque}})\left\{[I_{3\times 3} + \text{Diag}(\boldsymbol{f}_{\text{torque}})]\hat{\boldsymbol{T}}^c_{\text{com}}\right.\\ &\quad \left.+\boldsymbol{b}_{\text{torque}} + \boldsymbol{w}_{\text{torque}}\right\}\end{aligned} \tag{70}$$

and the impulsive Δv in Eq. (57) is given by

$$\begin{aligned}\Delta\boldsymbol{v}^i_{\text{act}}\left(\Delta\hat{\boldsymbol{v}}^c_{\text{com}}, \boldsymbol{q}^i_c, \boldsymbol{p}_{\Delta v}\right) &= \mathcal{T}\left(\boldsymbol{q}^c_i\right)\delta\mathcal{T}(\boldsymbol{\epsilon}_{\Delta v})\left\{[I_{3\times 3} + \text{Diag}(\boldsymbol{f}_{\Delta v})]\Delta\hat{\boldsymbol{v}}^c_{\text{com}}\right.\\ &\quad \left.+\boldsymbol{b}_{\Delta v} + \Delta\boldsymbol{w}_{\Delta v}\right\}\end{aligned} \tag{71}$$

where the actuator commands $\Delta\hat{\boldsymbol{v}}^c_{\text{com}}$ and $\hat{\boldsymbol{T}}^c_{\text{com}}$ generated by the flight computer are given in Eqs. (112), (116), and (117), and the covariance of the actuation noise is given by

$$E\left[\boldsymbol{w}_{\text{torque}}(t)\boldsymbol{w}^T_{\text{torque}}(t')\right] = S_{w_{\text{torque}}}\delta(t - t') \tag{72}$$

$$E\left[\Delta\boldsymbol{w}_{\Delta v}(t_j)\Delta\boldsymbol{w}^T_{\Delta v}(t_{j'})\right] = S_{\Delta w_{\Delta v}}(t_j)\delta_{jj'} \tag{73}$$

These simple models can accommodate a wide range of actuator errors and uncertainties and are also very useful in conducting trajectory control and navigation analysis.

C. Sensor Models

The gyro model is based on a package of three orthogonal strapdown gyros, each of which measures the component of the chaser angular velocity along its input axis. The gyro parameters are scale-factor error $\boldsymbol{f}_{\text{gyro}}$, misalignment error $\boldsymbol{\epsilon}_{\text{gyro}}$, and bias error $\boldsymbol{b}_{\text{gyro}}$, or $\boldsymbol{p}_{\text{gyro}} = (\boldsymbol{f}_{\text{gyro}}, \boldsymbol{\epsilon}_{\text{gyro}}, \boldsymbol{b}_{\text{gyro}})^T$. The output of the gyro model is

$$\tilde{\boldsymbol{\omega}}^c_c = \delta\mathcal{T}(\boldsymbol{\epsilon}_{\text{gyro}})[I_{3\times 3} + \text{Diag}(\boldsymbol{f}_{\text{gyro}})]\boldsymbol{\omega}^c_c + \boldsymbol{b}_{\text{gyro}} + \boldsymbol{\eta}_{\text{gyro}} \tag{74}$$

where the covariance of the gyro noise (angular random walk) is

$$E\left[\boldsymbol{\eta}_{\text{gyro}}(t)\boldsymbol{\eta}^T_{\text{gyro}}(t')\right] = S_{\eta_{\text{gyro}}}\delta(t - t') \tag{75}$$

The star camera is used to measure the three-axis orientation of the chaser vehicle. The model contains an uncertainty in the alignment of the star-camera frame with respect to the chaser frame $\boldsymbol{\epsilon}_{\text{starcam}}$. This uncertainty is the star-camera parameter, $\boldsymbol{p}_{\text{starcam}} = \boldsymbol{\epsilon}_{\text{starcam}}$. The output of the star-camera model is a quaternion of the form

$$\tilde{\boldsymbol{q}}^i_{\text{starcam}} = \delta\boldsymbol{q}(\boldsymbol{\nu}_{\text{starcam}}) \otimes \delta\boldsymbol{q}(\boldsymbol{\epsilon}_{\text{starcam}}) \otimes \bar{\boldsymbol{q}}^c_{\text{starcam}} \otimes \boldsymbol{q}^i_c \tag{76}$$

where the covariance of the measurement noise $\boldsymbol{\nu}_{\text{starcam}}$ is given by

$$E\left[\boldsymbol{\nu}_{\text{starcam}}(t_k)\boldsymbol{\nu}^T_{\text{starcam}}(t_{k'})\right] = R_{\nu_{\text{starcam}}}\delta_{kk'} \tag{77}$$

The optical tracking camera is used by the chaser to image and track the resident space object or a preselected feature of the resident space object. The instrument effectively provides line-of-sight information by measuring the object's pixel location in the camera focal plane. The model contains an uncertainty in the alignment of the optical tracking camera frame with respect to the chaser frame $\boldsymbol{\epsilon}_{\text{optrk}}$. This uncertainty is the optical tracking camera parameter, $\boldsymbol{p}_{\text{optrk}} = \boldsymbol{\epsilon}_{\text{optrk}}$. The focal plane measurements $\tilde{\boldsymbol{l}}^{\text{optrk}}$ from the optical tracking camera[18] are given by

$$\tilde{\boldsymbol{l}}^{\text{optrk}} = \begin{pmatrix} l_x/l_z \\ l_y/l_z \end{pmatrix} + \boldsymbol{\nu}_{\text{optrk}} \tag{78}$$

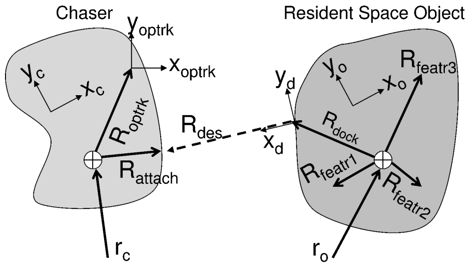

Fig. 2 Coordinate frames and feature locations in the chaser and resident space object spacecraft.

where the relative position of the object or object feature, $\boldsymbol{l}^{\text{optrk}}_{\text{los}} = [l_x, l_y, l_z]^T$, in the camera frame is given by

$$\begin{aligned}\boldsymbol{l}^{\text{optrk}}_{\text{los}} &= \delta\mathcal{T}(\boldsymbol{\epsilon}_{\text{optrk}})\bar{\mathcal{T}}^c_{\text{optrk}}\left\{\mathcal{T}\left(\boldsymbol{q}^i_c\right)\left[\boldsymbol{r}^i_o + \mathcal{T}\left(\boldsymbol{q}^o_i\right)\boldsymbol{R}^o_{\text{featr}}\right]\right.\\ &\quad \left.-\left[\mathcal{T}\left(\boldsymbol{q}^i_c\right)\boldsymbol{r}^i_c + \boldsymbol{R}^c_{\text{optrk}}\right]\right\}\end{aligned} \tag{79}$$

The feature position $\boldsymbol{R}^o_{\text{featr}}$ and optical tracking camera position $\boldsymbol{R}^c_{\text{optrk}}$ are indicated in Fig. 2. The feature position is set to zero for center-of-mass tracking. The covariance of the camera measurement noise is

$$E\left[\boldsymbol{\nu}_{\text{optrk}}(t_k)\boldsymbol{\nu}^T_{\text{optrck}}(t_{k'})\right] = R_{\nu_{\text{optrk}}}\delta_{kk'} \tag{80}$$

IV. Monte Carlo Flight Algorithm Models for Rendezvous

Flight algorithms are typically developed from design models, which are in turn derived from complex truth models. Remember that a caret refers to parameters, variables, and functions associated with flight algorithms, not estimates or expected values.

The navigation state is defined by an $\hat{n}$-dimensional vector ($\hat{n} \leq n$):

$$\hat{\boldsymbol{x}} = (\hat{\boldsymbol{x}}_o, \hat{\boldsymbol{x}}_c, \hat{\boldsymbol{p}})^T \tag{81}$$

It consists of 13 resident space object states, 10 chaser states, and 24 parameter states:

$$\hat{\boldsymbol{x}}_o = \left(\hat{\boldsymbol{r}}^i_o, \hat{\boldsymbol{v}}^i_o, \hat{\boldsymbol{q}}^i_o, \hat{\boldsymbol{\omega}}^o_o\right)^T \tag{82}$$

$$\hat{\boldsymbol{x}}_c = \left(\hat{\boldsymbol{r}}^i_c, \hat{\boldsymbol{v}}^i_c, \hat{\boldsymbol{q}}^i_c\right)^T \tag{83}$$

$$\hat{\boldsymbol{p}} = (\hat{\boldsymbol{p}}_{\text{gyro}}, \hat{\boldsymbol{p}}_{\text{starcam}}, \hat{\boldsymbol{p}}_{\text{optrk}}, \hat{\boldsymbol{p}}_{\Delta v})^T \tag{84}$$

Note that the chaser state does not contain angular velocity because the attitude model (given next) operates in model replacement mode,[18] where gyro measurements replace the Euler equations that would otherwise model the dynamics of the angular velocity state. This is not the case for the resident space object because gyro data for the object are not available.

A. Navigation Algorithm

The navigation state propagation equations are given by

$$\dot{\hat{\boldsymbol{r}}}^i_o = \hat{\boldsymbol{v}}^i_o \tag{85}$$

$$\dot{\hat{\boldsymbol{v}}}^i_o = \hat{\boldsymbol{F}}^i_{\text{grav}_o}\left(\hat{\boldsymbol{r}}^i_o\right)\big/\hat{m}_o \tag{86}$$

$$\dot{\hat{\boldsymbol{q}}}^i_o = \tfrac{1}{2}\hat{\boldsymbol{\omega}}^o_o \otimes \hat{\boldsymbol{q}}^i_o \tag{87}$$

$$\dot{\hat{\boldsymbol{\omega}}}^o_o = \hat{I}^{-1}_o\left[\hat{\boldsymbol{T}}^o_{\text{grav}_o}\left(\hat{\boldsymbol{r}}^i_o, \hat{\boldsymbol{q}}^i_o\right) - \hat{\boldsymbol{\omega}}^o_o \times \hat{I}_o\hat{\boldsymbol{\omega}}^o_o\right] \tag{88}$$

$$\dot{\hat{\boldsymbol{r}}}^i_c = \hat{\boldsymbol{v}}^i_c \tag{89}$$

$$\dot{\hat{\boldsymbol{v}}}^i_c = \hat{\boldsymbol{F}}^i_{\text{grav}_c}\left(\hat{\boldsymbol{r}}^i_c\right)\big/\hat{m}_c \tag{90}$$

$$\dot{\hat{\boldsymbol{q}}}{}_c^i = \tfrac{1}{2}\left[\tilde{\boldsymbol{\omega}}_c^c + \Delta\hat{\boldsymbol{\omega}}_{\text{comp}}(\boldsymbol{p}_{\text{gyro}})\right] \otimes \hat{\boldsymbol{q}}_c^i \tag{91}$$

$$\dot{\hat{p}}_i = -(\hat{p}_i/\hat{\tau}_i) \qquad i = 1, 2, 3, \ldots, \hat{n}_p \tag{92}$$

The gravitational forces on the chaser and the object $\boldsymbol{F}^i_{\text{grav}_c}$ and $\boldsymbol{F}^i_{\text{grav}_o}$, are point-mass plus J_2 gravity models,[16] and gravity-gradient torques $\boldsymbol{T}^o_{\text{grav}_o}$ are based on point-mass gravity models.[17] The gyro compensation $\Delta\hat{\boldsymbol{\omega}}_{\text{comp}}$ corrects the gyro measurement data $\tilde{\boldsymbol{\omega}}_c^c$ using the model in Eq. (74) and the flight computer values of the gyro bias, scale factor, and misalignment:

$$\Delta\hat{\boldsymbol{\omega}}_{\text{comp}}(\boldsymbol{p}_{\text{gyro}}) = \{[\hat{\boldsymbol{\epsilon}}_{\text{gyro}}\times] - \text{Diag}(\hat{\boldsymbol{f}}_{\text{gyro}})\}\tilde{\boldsymbol{\omega}}_c^c - \hat{\boldsymbol{b}}_{\text{gyro}} \tag{93}$$

The navigation state covariance propagation equation is

$$\dot{\hat{P}} = [\hat{F}_{\hat{x}} + \hat{F}_{\tilde{y}}\hat{C}_{\hat{x}}]\hat{P} + \hat{P}[\hat{F}_{\hat{x}} + \hat{F}_{\tilde{y}}\hat{C}_{\hat{x}}]^T + \hat{F}_{\tilde{y}}\hat{S}_{\eta_{\text{gyro}}}\hat{F}_{\tilde{y}}^T + \hat{S}_w \tag{94}$$

where the partial derivatives $\hat{C}_{\hat{x}}$, $\hat{F}_{\hat{x}}$, and $\hat{F}_{\tilde{y}}$ and the state process noise covariance $\hat{S}_w$ are given in Eqs. (A1–A9) in the Appendix. $\hat{S}_{\eta_{\text{gyro}}}$ is the flight computer's value of the covariance of the gyro angular random walk $S_{\eta_{\text{gyro}}}$ in Eq. (75).

Note that "modified" states[19] are used in deriving the preceding covariance propagation equation. The four-dimensional quaternion states $\hat{\boldsymbol{q}}_c^i$ and $\hat{\boldsymbol{q}}_o^i$ are replaced by three-dimensional rotation vector states $\hat{\boldsymbol{\theta}}_c^c$ and $\hat{\boldsymbol{\theta}}_o^o$ and the quaternion kinematics are replaced by the Bortz equation.[18] Thus, quaternion errors $\delta\hat{\boldsymbol{q}}$ are replaced by three-dimensional small-angle rotation vectors $\delta\hat{\boldsymbol{\theta}}$, and the linearized Bortz equation is used to derive the attitude covariance propagation equations.

All of the navigation state update equations

$$\delta\hat{\boldsymbol{x}}_k = \hat{K}(t_k)[\tilde{z}_k - \hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}_k, t_k)] \tag{95}$$

are additive:

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \delta\hat{\boldsymbol{x}}_k \tag{96}$$

with the exception of the attitude updates, which are implemented as small quaternion rotations,

$$\left(\hat{\boldsymbol{q}}_o^i\right)^+ = \delta\boldsymbol{q}\left(\delta\hat{\boldsymbol{\theta}}_o^o\right) \otimes (\hat{\boldsymbol{q}}_o^i)^- \tag{97}$$

$$\left(\hat{\boldsymbol{q}}_c^i\right)^+ = \delta\boldsymbol{q}\left(\delta\hat{\boldsymbol{\theta}}_c^c\right) \otimes \left(\hat{\boldsymbol{q}}_c^i\right)^- \tag{98}$$

When processing tracking camera data, $\tilde{z}_k$ is a vector of the focal plane measurements of the object or feature being tracked. The flight computer estimate of the focal plane measurements $\hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}_k, t_k)$ is based on the model in Eq. (78).

When processing star-camera data, $\tilde{z}_k$ is a derived measurement[18] calculated from

$$\begin{pmatrix} \frac{1}{2}\tilde{\boldsymbol{z}}_k \\ 1 \end{pmatrix} = \tilde{\boldsymbol{q}}^i_{\text{starcam}} \otimes \hat{\boldsymbol{q}}_i^c \otimes \delta\boldsymbol{q}(\hat{\boldsymbol{\epsilon}}_{\text{starcam}}) \otimes \bar{\boldsymbol{q}}_c^{\text{starcam}} \tag{99}$$

where for an extended Kalman filter implementation $\tilde{z}$ is effectively the residual to be processed by the Kalman filter. For a standard Kalman filter implementation, the derived measurement is calculated from

$$\begin{pmatrix} \frac{1}{2}\tilde{\boldsymbol{z}}_k \\ 1 \end{pmatrix} = \tilde{\boldsymbol{q}}^i_{\text{starcam}} \otimes \bar{\boldsymbol{q}}_i^c \otimes \bar{\boldsymbol{q}}_c^{\text{starcam}} \tag{100}$$

and the flight computer estimate of the derived star-camera measurement is

$$\hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}_k, t_k) = \hat{\mathcal{T}}^c_{\text{starcam}}\hat{\boldsymbol{\theta}}_c^c + \hat{\boldsymbol{\epsilon}}_{\text{starcam}} \tag{101}$$

where $\hat{\boldsymbol{\epsilon}}_{\text{starcam}}$ is the flight computer value of the star-camera misalignment.

The navigation state covariance update equation is given by

$$\hat{P}\left(t_k^+\right) = [I - \hat{K}(t_k)\hat{H}_{\hat{x}}(t_k)]\hat{P}\left(t_k^-\right)[I - \hat{K}(t_k)\hat{H}_{\hat{x}}(t_k)]^T + \hat{K}(t_k)\hat{R}_\nu(t_k)\hat{K}^T(t_k) \tag{102}$$

where the Kalman gain $\hat{K}(t_k)$ is given in Eq. (17). The measurement sensitivity matrices $\hat{H}_{\hat{x}}$ and the measurement covariance matrices $\hat{R}_\nu$ for the star camera and the optical tracking camera are given in Eqs. (A10–A14).

Finally, when an impulsive maneuver Δv is used to correct the chaser velocity, a correction must be made to the navigation state:

$$\left(\hat{\boldsymbol{v}}_c^i\right)^{+c} = \left(\hat{\boldsymbol{v}}_c^i\right)^{-c} + \Delta\hat{\boldsymbol{v}}^i_{\text{act}}\left(\Delta\hat{\boldsymbol{v}}^c_{\text{com}}, \hat{\boldsymbol{q}}_c^i, \hat{\boldsymbol{p}}_{\Delta v}\right) \tag{103}$$

where

$$\Delta\hat{\boldsymbol{v}}^i_{\text{act}}\left(\Delta\hat{\boldsymbol{v}}^c_{\text{com}}, \hat{\boldsymbol{q}}_c^i, \hat{\boldsymbol{p}}_{\Delta v}\right) = \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}_i^c\right)\delta\hat{\mathcal{T}}(\hat{\boldsymbol{\epsilon}}_{\Delta v})\left\{[I_{3\times 3} + \text{Diag}(\hat{\boldsymbol{f}}_{\Delta v})] \times \Delta\hat{\boldsymbol{v}}^c_{\text{com}} + \hat{\boldsymbol{b}}_{\Delta v}\right\} \tag{104}$$

The maneuver command $\Delta\hat{\boldsymbol{v}}^c_{\text{com}}$ is given in Eqs. (112) and (116), and $\hat{\boldsymbol{f}}_{\Delta v}$, $\hat{\boldsymbol{\epsilon}}_{\Delta v}$, and $\hat{\boldsymbol{b}}_{\Delta v}$ are the flight computer values of actuator scale factor, misalignment, and bias. The actuation noise covariance caused by (for example) actuator quantization error is simply

$$\hat{S}^{quant}_{\Delta w} = \text{Diag}\left(\left[\sigma^2_{\text{quant}_x}, \sigma^2_{\text{quant}_y}, \sigma^2_{\text{quant}_z}\right]\right) \tag{105}$$

When a correction is made to the navigation state, a correction is also made to the navigation state covariance matrix

$$\hat{P}\left(t_j^{+c}\right) = [I + \hat{D}_{\hat{x}}(t_j)]\hat{P}\left(t_j^{-c}\right)[I + \hat{D}_{\hat{x}}(t_j)]^T + \hat{S}^{\text{quant}}_{\Delta w}(t_j) \tag{106}$$

where the partial derivative $\hat{D}_{\hat{x}}$ is given in Eqs. (A15–A17).

B. Maneuver Targeting and Translation Control Algorithms

The maneuver targeting algorithm is based on the Clohessy–Wiltshire (CW) equations[20] and is referred to as the CW-targeting algorithm. Because CW targeting assumes two orbiting spacecraft are in close proximity and in near-circular orbits with no gravitational perturbations or differential drag, its validity will be dependent on the particular application. More sophisticated algorithms can be easily incorporated as needed. In a local-vertical/local-horizontal (lvlh) coordinate frame, the CW equations can be written as

$$\begin{pmatrix} \hat{\boldsymbol{R}}^{\text{lvlh}}(t_f) \\ \hat{\boldsymbol{V}}^{\text{lvlh}}(t_f) \end{pmatrix} = \begin{pmatrix} \Phi_{\text{rr}} & \Phi_{\text{rv}} \\ \Phi_{\text{vr}} & \Phi_{\text{vv}} \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{R}}^{\text{lvlh}}(t_o) \\ \hat{\boldsymbol{V}}^{\text{lvlh}}(t_o) \end{pmatrix} \tag{107}$$

where

$$\hat{\boldsymbol{R}}^{\text{lvlh}}(t_o) = \hat{\mathcal{T}}^i_{\text{lvlh}}\left(\hat{\boldsymbol{r}}_c^i - \hat{\boldsymbol{r}}_o^i\right) \tag{108}$$

$$\hat{\boldsymbol{V}}^{\text{lvlh}}(t_o) = \hat{\mathcal{T}}^i_{\text{lvlh}}\left[\hat{\boldsymbol{v}}_c^i - \hat{\boldsymbol{v}}_o^i - \hat{\boldsymbol{\omega}}^i_{\text{orb}} \times \left(\hat{\boldsymbol{r}}_c^i - \hat{\boldsymbol{r}}_o^i\right)\right] \tag{109}$$

$$\hat{\boldsymbol{\omega}}^i_{\text{orb}} = \hat{\boldsymbol{v}}_o^i \times \hat{\boldsymbol{r}}_o^i \Big/ \|\hat{\boldsymbol{r}}_o^i\|^2 \tag{110}$$

To achieve a desired relative position $\hat{\boldsymbol{R}}^{\text{lvlh}}_{\text{des}}$ at time t_f, the required $\Delta\boldsymbol{v}$ at time t_o is

$$\Delta\hat{\boldsymbol{v}}^{\text{lvlh}}_{\text{req}} = \Phi_{\text{rv}}^{-1}\left[\hat{\boldsymbol{R}}^{\text{lvlh}}_{\text{des}} - \Phi_{\text{rr}}\hat{\mathcal{T}}^i_{\text{lvlh}}\left(\hat{\boldsymbol{r}}_c^i - \hat{\boldsymbol{r}}_o^i\right)\right] - \hat{\mathcal{T}}^i_{\text{lvlh}}\left[\hat{\boldsymbol{v}}_c^i - \hat{\boldsymbol{v}}_o^i - \hat{\boldsymbol{\omega}}^i_{\text{orb}} \times \left(\hat{\boldsymbol{r}}_c^i - \hat{\boldsymbol{r}}_o^i\right)\right] \tag{111}$$

The $\Delta\boldsymbol{v}$ command that goes out to the actuators is the required $\Delta\boldsymbol{v}$ transformed to the chaser frame with compensation for known actuator errors:

$$\Delta\hat{\boldsymbol{v}}^c_{\text{com}} = \delta\hat{\mathcal{T}}(-\hat{\boldsymbol{\epsilon}}_{\Delta v})[I_{3\times 3} - \text{Diag}(\hat{\boldsymbol{f}}_{\Delta v})]\hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}_c^i\right)\hat{\mathcal{T}}_i^{\text{lvlh}}\Delta\hat{\boldsymbol{v}}^{\text{lvlh}}_{\text{req}} - \hat{\boldsymbol{b}}_{\Delta v} \tag{112}$$

A translation control algorithm is used for close-in proximity operations (such as inspection and docking) and employs a proportional-derivative (PD) controller to track a desired trajectory.

The desired trajectory is specified by the desired position of the chaser attach point $\hat{\boldsymbol{R}}^c_{\text{attach}}$, relative to a docking port location $\hat{\boldsymbol{R}}^o_{\text{dock}}$, on the resident space object as a function of time (see Fig. 2). At any given time, the required $\Delta\boldsymbol{v}$ is given by

$$\Delta\hat{\boldsymbol{v}}^i_{\text{req}} = K_r\left(\hat{\boldsymbol{r}}^i_c - \hat{\boldsymbol{r}}^i_{\text{des}}\right) + K_v\left(\hat{\boldsymbol{v}}^i_c - \hat{\boldsymbol{v}}^i_{\text{des}}\right) \tag{113}$$

where

$$\hat{\boldsymbol{r}}^i_{\text{des}} = \hat{\boldsymbol{r}}^i_o + \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^o_i\right)\left[\hat{\boldsymbol{R}}^o_{\text{dock}} + \hat{\mathcal{T}}\left(\bar{\boldsymbol{q}}^d_o\right)\hat{\boldsymbol{R}}^d_{\text{des}}\right] - \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^{\text{com}}_i\right)\hat{\boldsymbol{R}}^c_{\text{attach}} \tag{114}$$

$$\begin{aligned}\hat{\boldsymbol{v}}^i_{\text{des}} = {}& \hat{\boldsymbol{v}}^i_o + \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^o_i\right)\left\{\hat{\mathcal{T}}\left(\bar{\boldsymbol{q}}^d_o\right)\hat{\boldsymbol{V}}^d_{\text{des}} + \hat{\boldsymbol{\omega}}^o_o \times \left[\hat{\boldsymbol{R}}^o_{\text{dock}} + \hat{\mathcal{T}}\left(\bar{\boldsymbol{q}}^d_o\right)\hat{\boldsymbol{R}}^d_{\text{des}}\right]\right\} \\ & - \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^{\text{com}}_i\right)\left(\hat{\boldsymbol{\omega}}^c_{\text{com}} \times \hat{\boldsymbol{R}}^c_{\text{attach}}\right)\end{aligned} \tag{115}$$

Like the maneuver targeting algorithm, the $\Delta\boldsymbol{v}$ command that goes out to the actuators is the required $\Delta\boldsymbol{v}$ transformed to the chaser frame with compensation for known actuator errors:

$$\Delta\hat{\boldsymbol{v}}^c_{\text{com}} = \delta\hat{\mathcal{T}}(-\hat{\boldsymbol{\epsilon}}_{\Delta v})[I_{3\times 3} - \text{Diag}(\hat{\boldsymbol{f}}_{\Delta v})]\hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^i_c\right)\Delta\hat{\boldsymbol{v}}^i_{\text{req}} - \hat{\boldsymbol{b}}_{\Delta v} \tag{116}$$

C. Rotational Control and Pointing Algorithms

Rotational control is achieved through a PD control law

$$\hat{T}^c_{\text{com}} = \hat{K}_\theta \Delta\hat{\boldsymbol{\theta}}^c_{\text{com}} + \hat{K}_\omega\left(\hat{\boldsymbol{\omega}}^c_c - \hat{\boldsymbol{\omega}}^c_{\text{com}}\right) \tag{117}$$

where at medium and long range, $r > 100$ m, the pointing algorithm commands the orientation of the chaser such that the boresight of the optical tracking camera is trained on the resident space object. The angular velocity command is based on the relative position and velocity of the object.

$$\Delta\hat{\boldsymbol{\theta}}^c_{\text{com}} = \frac{\hat{\boldsymbol{R}}^c_{\text{optrk}}}{\left\|\hat{\boldsymbol{R}}^c_{\text{optrk}}\right\|} \times \frac{\hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^i_c\right)\left(\hat{\boldsymbol{r}}^i_o - \hat{\boldsymbol{r}}^i_c\right)}{\left\|\hat{\boldsymbol{r}}^i_o - \hat{\boldsymbol{r}}^i_c\right\|} \tag{118}$$

$$\hat{\boldsymbol{\omega}}^c_{\text{com}} = \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^i_c\right)\left[\frac{\left(\hat{\boldsymbol{r}}^i_o - \hat{\boldsymbol{r}}^i_c\right) \times \left(\hat{\boldsymbol{v}}^i_o - \hat{\boldsymbol{v}}^i_c\right)}{\left\|\hat{\boldsymbol{r}}^i_o - \hat{\boldsymbol{r}}^i_c\right\|^2}\right] \tag{119}$$

At short ranges, $r < 100$ m, such as for inspection, berthing, and/or docking, the chaser is commanded to match the attitude and attitude rate of the resident space object:

$$\hat{\boldsymbol{q}}^i_{\text{com}} = \hat{\boldsymbol{q}}^o_{\text{des}} \otimes \hat{\boldsymbol{q}}^i_o \tag{120}$$

$$\begin{pmatrix}\Delta\hat{\boldsymbol{\theta}}^c_{\text{com}} \\ 1\end{pmatrix} = \hat{\boldsymbol{q}}^i_{\text{com}} \otimes \hat{\boldsymbol{q}}^c_i \tag{121}$$

$$\hat{\boldsymbol{\omega}}^c_{\text{com}} = \hat{\mathcal{T}}\left(\hat{\boldsymbol{q}}^o_{\text{des}}\right)\hat{\boldsymbol{\omega}}^o_o \tag{122}$$

where $\hat{\boldsymbol{q}}^o_{\text{des}}$ is the offset orientation of the chaser with respect to the resident space object.

V. Rendezvous Analysis and Autonomous Mission Planning

The Monte Carlo rendezvous simulation is a fairly complex closed-loop GN&C simulation with pointing, maneuver targeting, attitude control, and navigation algorithms. Unfortunately, it can take many hours to execute the required number of samples. This is when linear covariance programs can be most useful.

This section contains two simple example applications of a rendezvous LinCov program. The program is derived using the methods of Sec. II and linearized versions of the Monte Carlo models and flight algorithms presented in Sec. III and IV. The first example illustrates the potential of using LinCov for autonomous onboard mission planning, and the second example illustrates the potential of LinCov for rendezvous analysis.

For both examples, two vehicles are in low-Earth near-circular orbits. A resident space object is in a passive near-lvlh orientation, and the chaser is actively controlling its relative attitude and position. The chaser flight computer processes continuous strap-down gyro measurements and discrete star-camera measurements every 10 s. Relative navigation is achieved using the optical tracking camera and by processing focal plane measurements of an image of the resident space object every 10 s. Translation control is achieved with short maneuvers modeled as impulsive burns, and attitude control is achieved with momentum wheels. Model parameters are shown in Table 1.

Table 1 Summary of rendezvous mission errors and uncertainties

Error	Comment	Value
	Initial condition error 3 σ	
Attitude and attitude rate dispersion/knowledge error	Per axis	0.1 rad, 0.01 mrad/s
Position and velocity dispersion/knowledge error	Downrange	300 m, 0.03 m/s
	Radial	30 m, 0.3 m/s
	crosstrack	300 m, 0.3 m/s
	Unmodeled disturbances 3 σ	
Rotational disturbances	Per axis	0.05 rad/rev
Translational disturbances	Downrange	100 m/rev
	Sensor error 3 σ	
Gyro error	Drift rate	3.0 deg/h/axis
	Scale factor	300 ppm/axis
	Misalignment	0.3 mrad/axis
	Random walk	0.05 mrad/$\sqrt{\text{s}}$
Star-camera error	Misalignment	1 mrad/axis
	noise	1 mrad/axis
Optical tracking camera error	Misalignment	1 mrad/axis
	noise	1 mrad/axis
	Actuator error 3 σ	
Maneuver execution error	Directional	1 mrad/axis
	bias	1 mm/s
	Scale factor	1000 ppm
	Random	0.1 mm/s
Torque execution error	Directional	1 mrad/axis
	bias	0.1 mN · mm
	scale factor	100 ppm

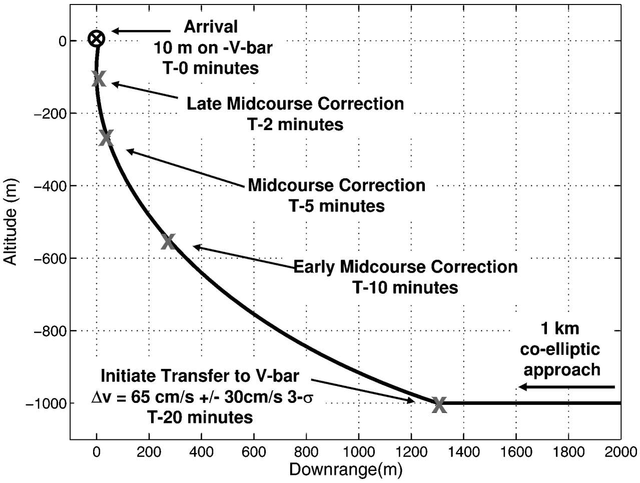

Fig. 3 Nominal approach is a 1-km coelliptic trajectory followed by a maneuver at a downrange position of 1300 m that transfers the chaser to a location 10 m in front (on the v-bar) of the resident space object in 20 min. A midcourse correction maneuver at T − 2 min, T − 5 min, and T − 10 min is to be selected autonomously by the onboard flight computer.

A. Onboard Mission Planning

In the first example, the chaser does a fast transfer from a 1-km coelliptic closing trajectory to a position 10 m in front (on the v-bar) of the resident space object in 20 min. The nominal trajectory is shown in Fig. 3.

When the chaser arrives at the 10-m v-bar position, there is a requirement that the relative position dispersion must be no greater than ±4-m 3 σ. To meet this requirement, the computer must determine if a midcourse maneuver is required and, if it is required,

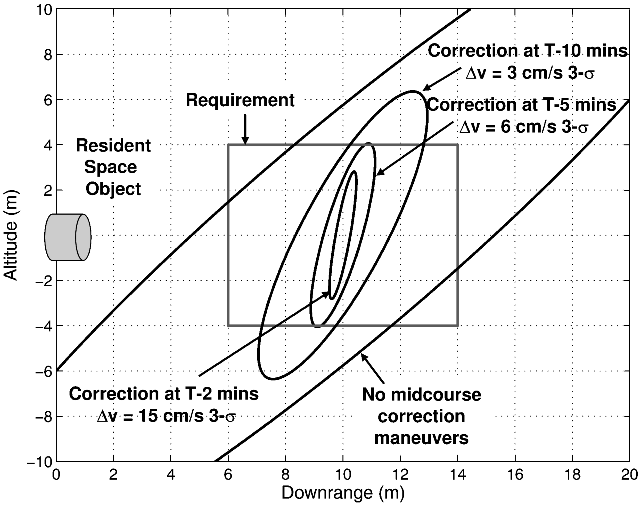

Fig. 4 Arrival dispersion ellipses as a function midcourse maneuver location are shown. The largest ellipse is the case without a midcourse maneuver. The maneuver with the lowest Δv that meets the dispersion requirement is at T − 5 min.

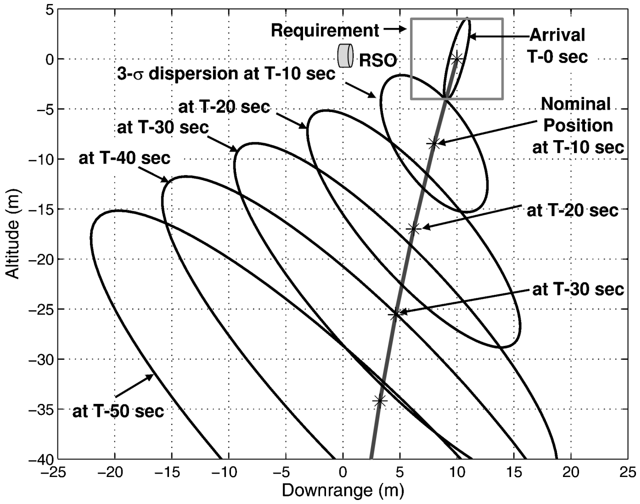

Fig. 5 Nominal trajectory and the dispersion ellipses for the final 60 s of the approach are shown. The midcourse maneuver is at T − 5 min.

select the best location to perform the maneuver. Typically, a late midcourse maneuver will achieve the desired dispersion requirement because the effects of navigation error and maneuver execution error are minimal, but the late maneuver will also require more Δv. An early midcourse maneuver will use a small Δv but might not meet the dispersion requirement if the maneuvering system is imprecise or if a good navigation solution is unavailable early in the final approach. Thus, the flight computer is used to autonomously determine which of the three maneuver locations shown in Fig. 3 is best by using linear covariance techniques. The entire process requires less than 1 min of CPU time on an Intel Pentium M 1.69-GHz processor.

Figure 4 shows the resulting 3-σ dispersion ellipses centered at the final 10-m v-bar position. The associated maneuver Δv is also shown. Notice first that without a midcourse correction the dispersions greatly exceed the dispersion requirement. A late midcourse maneuver (T − 2 min) clearly satisfies the dispersion requirement but requires 15-cm/s 3 σ of Δv. If the maneuver occurs slightly earlier (T − 5 min), the accuracy requirement can still be met but only requires 6-cm/s 3 σ. This is the maneuver selected by the flight computer. Figures 5–7 show time histories of the trajectory dispersions, relative position navigation errors, and chaser attitude uncertainties for the selected midcourse maneuver.

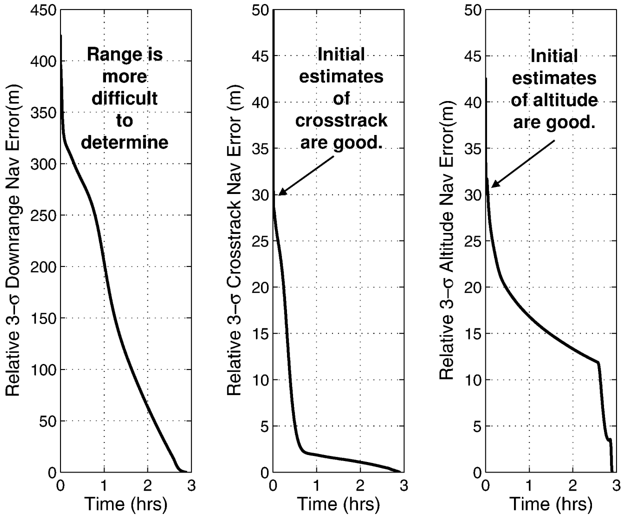

Fig. 6 Relative downrange, crosstrack, and altitude navigation errors for the entire approach. The angle measurements from the optical tracking camera allow for good crosstrack and altitude determination. Range is more difficult to determine.

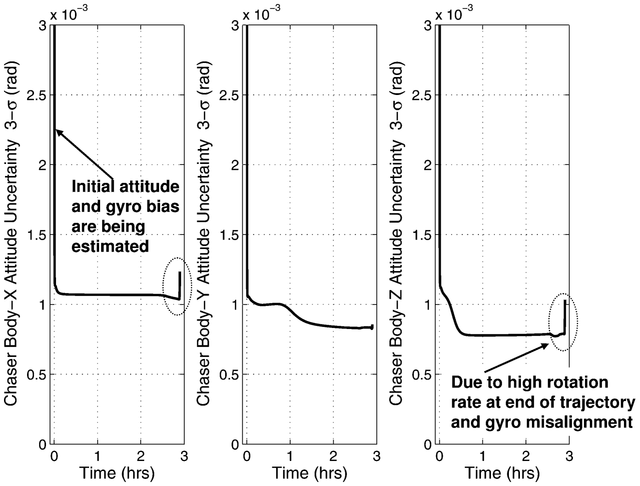

Fig. 7 Attitude uncertainties for the entire approach. Initial attitude errors and gyro biases are estimated quickly. Large rotation rates at the end of the trajectory (5 deg/s) coupled with gyro orthogonality errors produce the attitude error jump at the end of the trajectory.

B. Rendezvous Mission Analysis

In the second example, the chaser does a fast transfer from the 1-km coelliptic to a point 25 m in front (on the v-bar) of the resident space object. After it arrives on the v-bar, it begins a 2-cm/s approach toward the object. For this example, the LinCov program is used to determine how well the orientation and position of a resident space object relative to the chaser can be determined as the chaser approaches the object from the 25-m v-bar position. All initial conditions, random disturbance levels, and sensor parameters errors are the same as in the preceding example, except for the optical tracking camera.

For this example, we assumed the camera is able to recognize and track three features of the resident space object as illustrated in Fig. 2. The three features are assumed to be separated by 20 cm and form an isosceles triangle. The angle between the line of sight and the normal to the triangle is 55 deg. Focal plane measurements of the three features are taken every 60 s, and the camera measurement errors are the same as the preceding example.

Three different levels of feature location knowledge error, 0.0 cm (perfect knowledge), 0.1 cm, and 1-cm 3 σ, were examined. Figure 8 shows the results of this analysis. Relative attitude errors are 0.015-rad 3 σ at the end of the approach for both the perfect and the

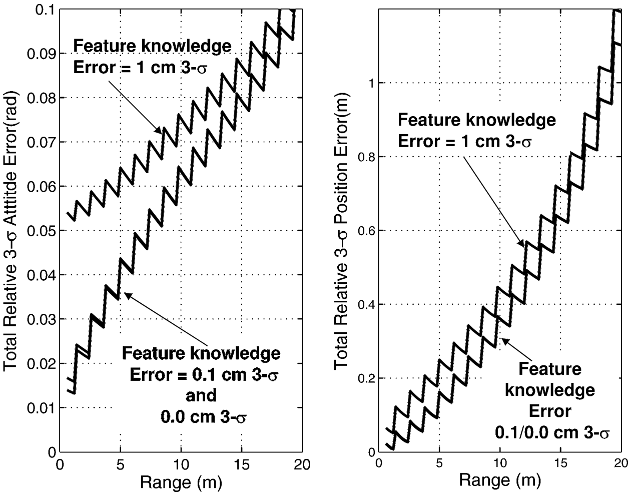

Fig. 8 Total relative attitude and total relative position errors as a function of downrange position and feature position knowledge error. The data for 0.0-cm 3-σ feature location knowledge error (i.e., perfect knowledge) and 0.1-cm 3 σ are nearly identical. Data are shown from 20-m range to 0.6-m range.

0.1-cm 3-σ feature location errors. With a 1-cm 3-σ location error, the relative attitude is 0.055-rad 3-σ at the end of the approach. Relative position error is not as sensitive to the feature location knowledge error, and the error is roughly 4% of the range. This analysis takes approximately one minute of CPU time on an Intel Pentium M 1.69-GHz processor.

VI. Conclusions

A new generalized methodology for developing GN&C linear covariance programs has been presented. The new approach accommodates continuous feedback control and inertial sensors, as well discrete/impulsive feedback control and noninertial sensors. The approach is applicable to a wide variety of GN&C problems and can be used to develop linear covariance programs to conduct analysis and solve a variety of problems.

Linear covariance programs have the potential to be used as onboard mission planning algorithms. This was clearly demonstrated by the results of a simple example in which the LinCov program computed the best maneuver correction location to meet a final position dispersion requirement while attempting to minimize Δv. In this respect, the LinCov techniques are unique in that they can account for the stochastic nature of the problem.

Linear covariance programs are also an excellent complement and/or alternative to traditional Monte Carlo programs for GN&C analysis. This was demonstrated by a simple example in which a LinCov program computed the variance of the relative attitude estimation errors. Many stochastic effects were considered in this problem, and a Monte Carlo approach would have required hundreds of simulations to obtain the same statistics. The LinCov simulation required only one simulation.

Appendix: Matrix Partial Derivatives

The propagation of the navigation state $\dot{\hat{\boldsymbol{x}}} = \hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}, \tilde{\boldsymbol{y}})$ is accomplished using Eqs. (85–92). The propagation of the navigation state covariance matrix in Eq. (94) is accomplished using the following partial derivatives and the state process noise covariance:

$$\hat{S}_\eta = \hat{S}_{\eta_{\text{gyro}}} \tag{A1}$$

$$\hat{C}_{\hat{x}} = \frac{\partial \tilde{\boldsymbol{\omega}}_c^c}{\partial \hat{\boldsymbol{x}}} = \begin{pmatrix} 0_{3\times 12} & 0_{3\times 9} & \partial\tilde{\boldsymbol{\omega}}_c^c / \partial\hat{\boldsymbol{p}}_{\text{gyro}} & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 9} \end{pmatrix} \tag{A2}$$

$$\hat{F}_{\tilde{y}} = \frac{\partial \hat{\boldsymbol{f}}}{\partial \tilde{\boldsymbol{\omega}}_c^c} = (0_{3\times 12} \quad 0_{3\times 6} \quad I_{3\times 3} \quad 0_{3\times 24})^T \tag{A3}$$

$$\hat{S}_w = \begin{pmatrix} \hat{S}_{w_o} & 0_{12\times 9} & 0_{12\times 24} \\ 0_{9\times 12} & \hat{S}_{w_c} & 0_{9\times 24} \\ 0_{24\times 12} & 0_{24\times 9} & -\text{Diag}\left(\left[\hat{\sigma}_{p_1}^2, \hat{\sigma}_{p_2}^2, \ldots, \hat{\sigma}_{p_{24}}^2\right]\right) \end{pmatrix} \tag{A4}$$

$$\hat{F}_{\hat{x}} = \frac{\partial \hat{\boldsymbol{f}}}{\partial \hat{\boldsymbol{x}}} = \begin{pmatrix} \hat{F}_{\hat{x}_o} & 0_{12\times 9} & 0_{12\times 24} \\ 0_{9\times 24} & \hat{F}_{\hat{x}_c} & 0_{9\times 24} \\ 0_{24\times 12} & 0_{24\times 9} & -\text{Diag}\left(\left[\frac{1}{\hat{\tau}_1}, \frac{1}{\hat{\tau}_2}, \ldots, \frac{1}{\hat{\tau}_{24}}\right]\right) \end{pmatrix} \tag{A5}$$

where

$$\hat{S}_{w_o} = \begin{pmatrix} 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & \hat{S}_{w_{\text{grav}}} + \hat{S}_{w_{\text{aero}}}^o & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & \sigma_{\dot{\omega}}^2 I_{3\times 3} \end{pmatrix} \tag{A6}$$

$$\hat{S}_{w_c} = \begin{pmatrix} 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & \hat{S}_{w_{\text{grav}}} + \hat{S}_{w_{\text{aero}}}^c & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \end{pmatrix} \tag{A7}$$

$$\hat{F}_{\hat{x}_o} =$$

$$\begin{pmatrix} 0_{3\times 3} & I_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ \dfrac{\hat{m}_o^{-1}\partial \boldsymbol{F}_{\text{grav}_o}^i}{\partial \hat{\boldsymbol{r}}_o^i} & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & -\left[\hat{\boldsymbol{\omega}}_o^o \times\right] & I_{3\times 3} \\ \dfrac{\hat{I}_o^{-1}\partial \boldsymbol{T}_{\text{grav}_o}^o}{\partial \hat{\boldsymbol{r}}_o^i} & 0_{3\times 3} & \dfrac{\hat{I}_o^{-1}\partial \boldsymbol{T}_{\text{grav}_o}^o}{\partial \hat{\boldsymbol{\theta}}_o^o} & \left\{\left[\left(I_o \hat{\boldsymbol{\omega}}_o^o\right) \times\right] - \left[\hat{\boldsymbol{\omega}}_o^o \times\right] I_o\right\} \end{pmatrix} \tag{A8}$$

$$\hat{F}_{\hat{x}_c} = \begin{pmatrix} 0_{3\times 3} & I_{3\times 3} & 0_{3\times 3} \\ \dfrac{\hat{m}_c^{-1}\partial \boldsymbol{F}_{\text{grav}_c}^i}{\partial \hat{\boldsymbol{r}}_c^i} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & -\left[\left(\tilde{\boldsymbol{\omega}}_c^c + \Delta\hat{\boldsymbol{\omega}}_{\text{comp}}\right) \times\right] \end{pmatrix} \tag{A9}$$

The measurement sensitivity matrix for the star camera is given by

$$\hat{H}_{\hat{x}}^{\text{starcam}} = \begin{pmatrix} 0_{3\times 12} & 0_{3\times 6} & \hat{\mathcal{T}}_{\text{starcam}}^c & 0_{3\times 9} & I_{3\times 3} & 0_{3\times 3} & 0_{3\times 9} \end{pmatrix} \tag{A10}$$

and the flight computer's value of star-camera measurement covariance is

$$\hat{R}_v^{\text{starcam}} = \text{Diag}\left(\left[\sigma_{\text{starcam}_x}^2, \sigma_{\text{starcam}_y}^2, \sigma_{\text{starcam}_z}^2\right]\right) \tag{A11}$$

The measurement sensitivity matrix for the optical tracking camera[18] is given by

$$\hat{H}_{\hat{x}}^{\text{optrk}} = \begin{pmatrix} 1/\hat{l}_z & 0 & -\hat{l}_x / \hat{l}_z^2 \\ 0 & 1/\hat{l}_z & -\hat{l}_y / \hat{l}_z^2 \end{pmatrix} \frac{\partial \hat{\boldsymbol{l}}_{\text{los}}^{\text{optrk}}}{\partial \hat{\boldsymbol{x}}} \tag{A12}$$

where

$$\begin{aligned} \frac{\partial \hat{\boldsymbol{l}}_{\text{los}}^{\text{optrk}}}{\partial \hat{\boldsymbol{x}}} = \Big(&\hat{\mathcal{T}}_{\text{optrk}}^i, 0_{3\times 3}, -\hat{\mathcal{T}}_{\text{optrk}}^o\left[\hat{\boldsymbol{R}}_{\text{featr}}^o \times\right], 0_{3\times 3}, -\hat{\mathcal{T}}_{\text{optrk}}^i, 0_{3\times 3}, \\ &\hat{\mathcal{T}}_{\text{optrk}}^c\left[\left(\hat{\boldsymbol{r}}_o^c + \hat{\boldsymbol{R}}_{\text{featr}}^c - \hat{\boldsymbol{r}}_c^c\right) \times\right], 0_{3\times 3}, 0_{3\times 3}, 0_{3\times 3}, \\ &\left[\left(\hat{\boldsymbol{r}}_o^{\text{optrk}} + \hat{\boldsymbol{R}}_{\text{featr}}^{\text{optrk}} - \hat{\boldsymbol{r}}_c^{\text{optrk}} - \hat{\boldsymbol{R}}_{\text{optrk}}^{\text{optrk}}\right) \times\right], 0_{3\times 9}\Big) \end{aligned} \tag{A13}$$

The flight computer's value of the tracking camera measurement covariance is

$$\hat{R}_v^{\mathrm{optrck}} = \mathrm{Diag}\left(\left[\sigma^2_{\mathrm{optrck}_x}, \sigma^2_{\mathrm{optrck}_y}\right]\right) \tag{A14}$$

The correction of the navigation state covariance matrix in Eq. (106) is accomplished using the following partial derivative:

$$\hat{D}_{\hat{x}} = \begin{pmatrix} 0_{12\times 12} & 0_{12\times 9} & 0_{12\times 24} \\ 0_{9\times 12} & \hat{D}_{\hat{x}_c} & \hat{D}_{\hat{x}_p} \\ 0_{24\times 12} & 0_{24\times 9} & 0_{24\times 24} \end{pmatrix} \tag{A15}$$

where

$$\hat{D}_{\hat{x}_c} = \begin{pmatrix} 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & \partial\Delta\hat{\boldsymbol{v}}^i_{\mathrm{act}} \big/ \partial\hat{\boldsymbol{\theta}}^c_c \\ 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \end{pmatrix} \tag{A16}$$

$$\hat{D}_{\hat{x}_p} = \begin{pmatrix} 0_{3\times 9} & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 9} \\ 0_{3\times 9} & 0_{3\times 3} & 0_{3\times 3} & \partial\Delta\hat{\boldsymbol{v}}^i_{\mathrm{act}} \big/ \partial\hat{\boldsymbol{p}}_{\Delta v} \\ 0_{3\times 9} & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 9} \end{pmatrix} \tag{A17}$$

References

[1] Rao, P. P., and Belt, S. C., "Conditional Performance Error Covariance Analysis for Commercial Titan Launch Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 2, 1991, pp. 398–404.

[2] Merel, M. H., and Mullin, F. J., "Analytic Monte Carlo Error Analysis," *Journal of Spacecraft and Rockets*, Vol. 5, No. 11, 1968, pp. 1304–1308.

[3] Porcelli, G., and Vogel, E., "Two-Impulse Transfer Error Analysis via Covariance Matrix," *Journal of Spacecraft and Rockets*, Vol. 17, No. 3, 1980, pp. 248–255.

[4] Purcell, E. W., and Barbery, T. B., "Dispersions Resulting from Atmospheric Variations," *Journal of Guidance, Control, and Dynamics*, Vol. 5, No. 8, 1968, pp. 1005–1007.

[5] LaFarge, R. A., and Baty, R. S., "Functional Dependence of Trajectory Dispersions on Initial Condition Errors," *Journal of Spacecraft and Rockets*, Vol. 31, No. 7, 1994, pp. 806–813.

[6] Zarchan, P., "Complete Statistical Analysis of Non-Linear Missile Guidance Systems—SLAM," *Journal of Guidance, Control, and Dynamics*, Vol. 2, No. 1, 1979, pp. 71–78.

[7] Crassidis, J. L., and Junkins, J. L., *Optimal Estimation of Dynamic Systems*, CRC Press, New York, 2004, pp. 308–310.

[8] Boone, J. N., "Generalized Covariance Analysis for Partially Autonomous Deep Space Missions," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 5, 1991, pp. 964–972.

[9] Mook, D. J., and Junkins, J. J., "Minimum Model Error Estimation for Poorly Modeled Dynamic Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 11, No. 3, 1988, pp. 256–261.

[10] Tabley, B. D., Schutz, B. E., and Born, G. E., *Statistical Orbit Determination*, Elsevier Academic Press, London, 2004, pp. 387–435.

[11] Gelb, A., *Applied Optimal Control*, MIT Press, Cambridge, MA, 1974, Chap. 7.

[12] Battin, R. H., *An Introduction to the Mathematics and Methods of Astrodynamics*, AIAA Education Series, AIAA, New York, 1987, pp. 681–884.

[13] Maybeck, P. S., *Stochastic Models, Estimation, and Control*, Academic Press, New York, 1979, pp. 325–340.

[14] Hehse, W., *Automated Rendezvous and Docking of Spacecraft*, Cambridge Univ. Press, New York, 2003, Appendix C.

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

[16] Kaplan, M. H., *Modern Spacecraft Dynamics and Control*, Wiley, New York, 1976, pp. 343–370.

[17] Wertz, J. R., *Spacecraft Attitude Determination and Control*, Kluwer Academic, London, 1978, pp. 566–570.

[18] Pittelkau, M. E., "Rotation Vector in Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 6, 2003, pp. 855–860.

[19] Lefferts, E. J., Markley, F. L., and Shuster, M. D., "Kalman Filtering for Spacecraft Estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 5, No. 5, 1982, pp. 417–429.

[20] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, McGraw–Hill, New York, 1997, pp. 348–365.