Distributed Control of Formation Flying Spacecraft Built on OA

Louis Breger[*], Philip Ferguson[†], Jonathan P. How,[‡] Stephanie Thomas[§]
Terence McLoughlin[¶] and Mark Campbell[‖]

ABSTRACT

This paper describes several tools and technologies that have been developed for future spacecraft formation flying missions. This includes algorithms to perform autonomous navigation and control, with new results presented for trajectory planning in highly elliptic orbits and fault detection for a distributed spacecraft system. The overall approach is embedded in the OA middleware developed by Princeton Satellite Systems, which provides a seamless integration of networking and process control with C/C++ software development. New results in this paper also demonstrate OA integrated with software that is running in hard real-time. A new multi-computer spacecraft formation flying testbed was created to simulate the performance of the full system. In particular, simulation results of an optimally initialized "recurring tetrahedron formation" on a highly elliptical orbit are shown to demonstrate both the functionality of the new testbed and the potential for creating fuel-efficient formation configurations relevant to future magnetospheric space science.

INTRODUCTION

Many future space missions will use coordinated satellites flying in formation to increase the resolution of the science data or achieve faster ground-track repeats.[1] For example, formation flying will be critical for creating large sparse-aperture optical and X-ray telescopes for space science and synthetic aperture radars for earth mapping. As discussed in [6], formation flying combines many component technologies, such as distributed relative navigation, autonomous control, and distributed fault-protection. While numerous algorithms have recently been designed to perform these various tasks, significant further testing is required to build confidence in the overall control system.[6] This paper presents extensions of several of these guidance and control algorithms to address the requirements for a new space science mission and discusses a testbed that was designed to assess the performance and robustness of the distributed fleet control system.

Existing formation flying testbeds are geared toward very specific research topics (e.g. GPS simulation, interferometer development, and architecture trade studies). This paper presents a new formation flying testbed to be used for testing and validating control and autonomy technologies. The new testbed uses the OA middleware designed by Princeton Satellite Systems (PSS) to implement flexible and highly distributed software architectures.[2] OA is used to facilitate process control, interprocess communication, and network communication. This, in turn, enables rapid, low effort, software reconfiguration through the use of common interfaces between previously disparate types of software communication.[2] To test and validate formation flying algorithms under realistic conditions, capabilities for utilizing real-time constraints, communication limits, and inserted failures have been designed into the system. In addition, the testbed is made up of software modules written in C and C++, which are languages commonly used to write flight software. As a result, the testbed software is highly adaptable and it can be used to simulate a variety of different missions, such as low Earth orbit formations (*e.g.,* TechSat21) and a cluster of satellites in a highly elliptic orbit (*e.g.,* Magnetospheric Multiscales (MMS) mission[14]). In both cases simulations were initially conducted on a single processor and then moved to a multi-processor system with computers in the testbed located at both Cornell and MIT.

Additionally, this paper presents several new technologies related to formation flying. One such area is drift-minimizing orbit initialization, which is vital for reducing the fuel necessary for formation flying. Several methods exist to perform this type of initial-

[*] Research Assistant, MIT AA, **lbreger@mit.edu**
[†] Research Assistant, MIT AA, **philf@mit.edu**
[‡] Associate Professor, MIT AA, **jhow@mit.edu**.
[§] Princeton Satellite Systems, **sjthomas@psatellite.com**
[¶] Research Assistant, Cornell Univ., **thm23@cornell.edu**
[‖] Assistant Professor, Cornell Univ., **mc288@cornell.edu**

ization for closely spaced formations, but the technique in Ref.[16] for large separations was only developed for a specific fleet configuration. This paper extends that technique to the general "drift-free" case and demonstrates the results on the new testbed. A further limitation in the current set of formation flying technologies is the linearized models which are used to propagate the relative dynamics in the trajectory planning.[3] For example, propagators written as a function of time[11] are only valid for reference orbit eccentricities of $e \lesssim 0.3$. A propagator that is valid for higher eccentricities is written as a function of true-anomaly, which complicates the planning process.[4] This paper presents a relative linear orbit propagator that is valid for all elliptical reference orbits and can be written as a function of time, thereby enabling fixed-time-step planning for high-eccentricity orbits. While this paper focuses on the control aspects, a companion paper addresses some of the navigation issues.[7]

Services provided by OA, such as dynamic process replacement and event-triggered communications have the potential to enhance fault detection and recovery in distributed systems.[2] An approach to fault protection that makes use of communication services provided by OA is developed and evaluated.

The MMS mission will be treated as a case study in this paper. MMS comes from a class of formation flying missions that use highly elliptic orbits ($e \approx 0.9$) and aim to study the magnetosphere.[13] The four satellites in the MMS formation will be widely separated and will be required to maintain a regular tetrahedron geometry for taking science data. These requirements taken together provide a challenging control problem that many of the following technologies could be applied to solving.

OA MIDDLEWARE

To develop robust architectures for distributed satellite systems, our research encompasses both algorithm development and software implementation. The fleet software is based on the ObjectAgent (OA) framework developed by PSS to increase the reconfigurability, modularity and reliability of the overall control system. With its emphasis on robustly handling communication between Agents that coordinate to complete complex tasks, the OA software provides a natural framework for developing these distributed autonomous GN&C algorithms. ObjectAgent extends the classical method for writing spacecraft software by using "software agents" as the basis of the system.

OA is a multi-threaded architecture developed for distributed systems. It uses message passing for thread communication and can run on any POSIX compliant operating system. The architecture has two tiers, PostOffice and Agent, and every instance of each is a separate POSIX thread. There may be any number of Agents in a PostOffice, any number of PostOffices on a processor, and any number of processors in the system. The Agent is the base unit for communication, and all OA messages are passed between Agents. An Agent knows its list of inputs and outputs and built-in functions enable it to hunt for inputs and automatically configure itself upon launch. In this sense, an ObjectAgent system is self-organizing.

The PostOffice enables seamless Agent communication throughout the OA system. In particular, each PostOffice manages a set of Agents, handles communication between different processors via TCP/IP or the user's choice of network protocol, and provides a framework for the dynamic creation of Agents. The PostOffice network is a fully routed network and can be reconfigured on the fly. Each link in the network can specify a separate network protocol, such as TCP/IP. Once the protocols are specified, Agent communication is transparent to the user; a Skill only needs to know the name of the data and the Agent providing or needing it.

There are several features of the OA architecture specifically related to the GN&C problem: (i) message passing; (ii) modularity; (iii) reconfigurability; and (iv) robustness to software faults.The flexible messaging architecture is a fundamental component of OA since it provides a reliable method for Agent-to-Agent communication both on a single processor and across networks. The inter-Agent messaging is handled by the PostOffice, such that the PostOffice provides distributed mutual exclusion services and efficient broadcast/discovery services to its Agents. Each message has both a creation and data timetag and input messages may be queued. Messages may be either point-to-point between Agents or broadcast by the PostOffice. An important result of using message passing is that the Agents can easily be monitored or commanded. An OA GUI is provided which can connect to an OA system using TCP/IP and allows users to view and record the messages for any PostOffice and send commands to any Agent.

OA was designed for distributed systems and development. Developers at different locations can write Agents and Skills and reliably combine them; the only components that must be managed are the data interfaces, as in any system. In the case of this project, students at MIT and Cornell develop separate Agents for various formation flying technologies and combine them in "Spacecraft" units on one PostOffice. Several computers at each campus are used for testing, with each computer hosting one or more Spacecraft or the simulation.

Another key feature of OA is run-time reconfigurability. Agents can be dynamically loaded and unloaded without shutting down the entire software system. An Agent can also be dynamically assigned new Skills. This is achieved by pre-compiling the new Agent or Skill into a library. The dynamic loading of Agents and PostOffices allows the Agent network to evolve over time and frees spacecraft operators from the expensive software patching process.

For OA to be useful for critical systems such as spacecraft control software, it must be robust to software failures. OA is extremely robust and fault tolerant on a Post-Office level, and moderately robust at the Agent level. However, a special feature called Agent sandboxing allows each Agent to be in its own memory space. Sandboxing a critical Agent will increase its robustness to failures equivalent to the PostOffice level in exchange for sacrificing some performance. Without sandboxing, run-time errors that occur within an Agent on one PostOffice will cause that PostOffice and all of its Agents to terminate. However, connected PostOffices are not be affected since PostOffices can gracefully recover from network errors.

Although fault tolerance is largely dependent on the user implementation, OA provides hooks wherever possible to notify Agents of problems, such as a notification that another Agent has died or returning an undeliverable message to its sender. Agents can also assess the state of the processor on which they operate to determine if they should pass tasks off to another Agent in the system.

In general, real-time applications in the context of control require a real-time operating system and a real-time network. OA was designed for us on any POSIX compliant operating system, not exclusively real-time operating systems, so OA does not make use of any specific hard real-time calls. Consequently, the degree of real-time guarantee for any application running in OA is dependent on the selected processor, operating system, and in the case of distributed control, network protocol. There are several options open to developers for enforcing soft timing constraints. First, individual OA threads may be set at any of the priority levels provided by the operating system. This allows higher level threads to block execution of lower threads. Second, each OA Agent thread can have a separate update rate. Although the thread may be aperiodic and not depend on any update rate, it may also update very rapidly in the case of a control thread or very slowly in the case of planning thread. OA's ability to operate in a real-time environment was demonstrated on a computer running real-time Linux under artificial loading conditions.

In summary, the OA software environment provides the following key attributes directly related to the overall goals of developing an autonomous GN&C system:

- Easily allows different software architectures (including different communication and computation schemes) to be defined and compared.
- Allows software to be quite flexible, with objects that can be dynamically loaded.
- Allows software to degrade gracefully.
- Allows transparency into software operation by monitoring Agent messages.

STRUCTURE OF THE TESTBED

The formation flying testbed is comprised of three types of OA agents: Simulation, Spacecraft, and Fault Detection. One agent performs the dynamics simulation for all spacecraft in the testbed, and there are spacecraft and fault detection agents for each vehicle in the simulated fleet. The spacecraft agent coordinates all spacecraft software functions (*e.g.,* control, planning, communication). Fault detection services have been removed to a separate agent for operational robustness.

The structure within agents is arranged to minimize the effort required to add capabilities to simulated spacecraft. Each agent is responsible for a very specific algorithmic task, with clearly defined inputs and outputs that effectively create "capability modules." The Simulation agent contains a propagation skill that inputs thrust control and outputs the new states of each spacecraft. Each spacecraft agent contains an Executive agent that is the control system for that spacecraft. The Executive agent receives state telemetry from the Simulation agent and returns thruster commands to the Simulation agent to be applied to the appropriate spacecraft in the dynamic simulation.

The physical structure of the testbed (*i.e.,* where the software runs) is highly variable. For developing new software, the agents could all be run in a single computer. However, a more realistic simulation

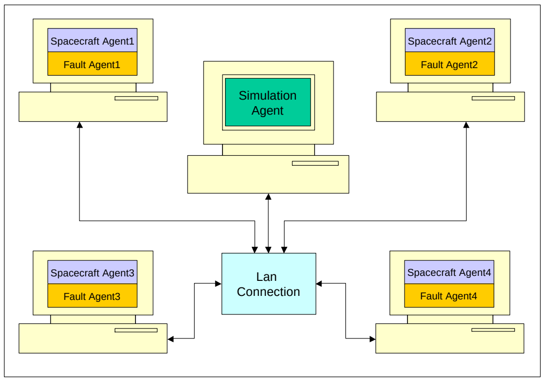

Fig. 1: Testbed Physical Structure

would place the software for each spacecraft on a separate computer, with an additional computer to simulate the vehicle motion and the environment (see Fig. 1). Switching between a stand-alone and distributed configuration is very simple in this case, because the OA interprocess and inter-computer communication are handled transparently by the same software interface, even when the computers are at different universities. The multi-computer research presented here was conducted between computers at Cornell and MIT.

FORMATION FLYING TECHNOLOGIES

This section describes the key technologies that have recently been integrated into the testbed.

Improvements in Initial Conditions

A number of future spacecraft formation flying missions will require spacecraft to maintain specified separations or relative geometries.[13] These requirements stem from the need to obtain scientific data simultaneously from widely separated locations or the need to take data in the same location at frequent intervals.[1]

To minimize control effort, orbits can be chosen that naturally prevent the spacecraft from separating. One type of drift-free orbit, a *passive aperture*, is based on the elimination of secular terms from Hill's equations.[8] The general solution to Hill's equations has six initial conditions that define a satellite's orbit relative to the origin of the Hill's frame, where x is the radial direction, y is the along-track direction, and z is the across-track direction. It can be seen that the only term contributing to secular drift is $-(3\dot{y}(0) + 6n_{\text{ref}}x(0))t$, where n_{ref} is the period of the osculating orbit and t is elapsed time. The condition to prevent spacecraft separation over time is

$$3\dot{y}(0) = -6n_{\text{ref}}x(0) \tag{1}$$

This approach works well for formations in circular orbits where the separation between spacecraft is on the order of 100 m. However, the accuracy of Hill's equations degrades as the inter-spacecraft separation is extended, therefore, choosing initial conditions based on Eq. 1 will no longer eliminate secular drift. Ref. 16 recently proposed an approach that extends the validity of Hill's equations to larger inter-spacecraft separations by adding a set of second order perturbations. This approach is derived for a specific solution to Hill's equations with initial conditions that restrict the formation to a projected circle in the y-z plane. In this case, the radius of the projected circle is the same for all satellites in the formation and the position of the satellites in the circle is chosen by an angular offset. This approach works well but is overly restrictive, as the formation geometry must be specified by only two initial conditions. The MMS mission will require a widely separated regular tetrahedron geometry to fulfill its science objectives, but the initial conditions available are not sufficient to describe a tetrahedron. To use the nonlinearity correction with MMS, the following extends the basic approach in Ref. [16] to a more general solution of Hill's equations:

$$\begin{aligned}
x(t) &= x(0)\cos(n_{\text{ref}}t) + \frac{\dot{x}(0)}{n_{\text{ref}}}\sin(n_{\text{ref}}t) \\
y(t) &= y(0) + \frac{2\dot{x}(0)}{n_{\text{ref}}}\big[\cos(n_{\text{ref}}t) - 1\big] - 2x(0)\sin(n_{\text{ref}}t) \\
z(t) &= z(0)\cos(n_{\text{ref}}t) + \frac{\dot{z}(0)}{n_{\text{ref}}}\sin(n_{\text{ref}}t)
\end{aligned} \tag{2}$$

where the relative orbit of each spacecraft in the Hill's frame is defined by five initial conditions, $x(0)$, $y(0)$, $z(0)$, $\dot{x}(0)$, and $\dot{z}(0)$. The Hill's solution in Eq. 2 can be used to define initial conditions corresponding to the corners of a tetrahedron, because the initial x, y, and z coordinates can be specified independently. Using Eq. 2, any initial velocity conditions will produce a recurring tetrahedron formation in the Hill's frame, but the nonlinearity correction derived herein can be applied to other geometries that may require specific initial velocities as well as positions. Optimizing the choice of initial conditions to account for other effects is discussed at the end of the subsection. Figure 2 shows a regular tetrahedron with a spacecraft at each vertex. Lines are shown to highlight the geometry. Tetrahedron examples used in this paper use the initial position conditions (kilometers) $\text{Sat}_1 = (0, 0, \frac{\sqrt{3}}{3}10)$, $\text{Sat}_2 = (0, \frac{1}{2}10, -\frac{\sqrt{3}}{6}10)$, $\text{Sat}_3 = (0, -\frac{1}{2}10, -\frac{\sqrt{3}}{6}10)$, and $\text{Sat}_4 = (\frac{\sqrt{6}}{3}10, 0, 0)$.

The nonlinearity correction only accounts for sec-

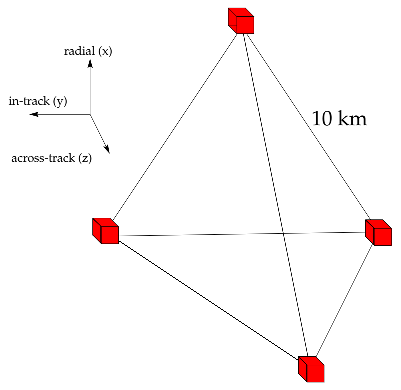

Fig. 2: Regular 10 km tetrahedron formation.

ond order effects. The fully nonlinear relative equations of motion (not shown) can be reduced in terms of eccentricity and nonlinearity into the following second order equations of motion[16]

$$\begin{aligned}\ddot{x} - 2n_{\text{ref}}\dot{y} - 3n_{\text{ref}}^2 x &= \epsilon\left[\frac{y^2}{2} + \frac{z^2}{2} - x^2\right] \qquad (3)\\ \ddot{y} - 2n_{\text{ref}}\dot{x} &= \epsilon xy\\ \ddot{z} - n_{\text{ref}}^2 \dot{z} &= \epsilon xz\end{aligned}$$

where $\epsilon = 3\mu/a_c^4$, μ is the gravitational parameter, and a_c is the semi-major axis of the reference orbit. Assuming the radial direction solution (and corresponding solutions in the in-track and cross-track directions)

$$\ddot{x} = \ddot{x}_{\text{h}} + \epsilon\ddot{x}_{\text{cn}}, \quad \dot{x} = \dot{x}_{\text{h}} + \epsilon\dot{x}_{\text{cn}}, \quad x = x_{\text{h}} + \epsilon x_{\text{cn}} \quad (4)$$

where $(\cdot)_{\text{h}}$ denotes a state of the Hill's equations and $(\cdot)_{\text{cn}}$ denotes a perturbation state. Eq. 3 can be rewritten as

$$\begin{aligned}\ddot{x}_{\text{cn}} - 2n_{\text{ref}}\dot{y}_{\text{cn}} - 3n_{\text{ref}}^2 x_{\text{cn}} &= (1/2)(y_{\text{h}}^2 + z_{\text{h}}^2 - 2x_{\text{h}}^2)\\ \ddot{y}_{\text{cn}} + 2n_{\text{ref}}\dot{x}_{\text{cn}} &= x_{\text{h}}y_{\text{h}}\\ \ddot{z}_{\text{cn}} + n_{\text{ref}}^2 z_{\text{cn}} &= x_{\text{h}}z_{\text{h}} \qquad (5)\end{aligned}$$

The right hand side of Eq. 5 can be found by substituting in the Hill's solution from Eq. 2, yielding Eq. 6 (see next page).

Treating the Hill's states as inputs, the perturbation states evolve as

$$\mathbf{X}_{\text{cn}}(\text{t}) = \Phi_{\text{h}}(\text{t})\mathbf{X}_{\text{cn}}(0) + \int_0^{\text{t}} \Phi_{\text{h}}(\text{t}-\tau)\text{B}\mathbf{u}_{\text{h}}(\tau)\text{d}\tau \quad (7)$$

where*

$$\mathbf{X}_{\text{cn}}(\text{t}) = \left[x_{\text{cn}}(t),\, y_{\text{cn}}(t),\, z_{\text{cn}}(t),\, \dot{x}_{\text{cn}}(t),\, \dot{y}_{\text{cn}}(t),\, \dot{z}_{\text{cn}}(t)\right]^{\text{T}}$$

*For brevity, n is used in place of n_{ref}, c in place of $\cos(nt)$, and s in place of $\sin(nt)$

$$\mathbf{u}_{\text{h}}(\text{t}) = \begin{bmatrix} \frac{y_{\text{h}}^2 + z_{\text{h}}^2 - 2x_{\text{h}}^2}{2} \\ x_{\text{h}}y_{\text{h}} \\ x_{\text{h}}z_{\text{h}} \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ I \end{bmatrix}$$

and

$$\Phi_{\text{h}}(t) = \begin{bmatrix} 4-3c & 0 & 0 & \frac{s}{n} & \frac{2(1-c)}{n} & 0 \\ 6(s-nt) & 1 & 0 & \frac{2(1-c)}{n} & \frac{4s-3nt}{n} & 0 \\ 0 & 0 & c & 0 & 0 & \frac{s}{n} \\ 3ns & 0 & 0 & c & 2s & 0 \\ -6n(1-c) & 0 & 0 & -2s & 4c-3 & 0 \\ 0 & 0 & -ns & 0 & 0 & c \end{bmatrix}$$

Evaluating Eq. 7 symbolically yields the solutions in Eqs. 15–20 (see Appendix). These six equations show that, similar to the case for the standard Hill's solution, only the along-track position, $y(t)$, has secular drift terms. The secular drift can be eliminated from the solution if the coefficient of that term is set equal to zero, which From Eq. 16 gives

$$-\frac{1}{2n^3}\Big(12n^4x(0)_{\text{cn}} + 6\dot{y}(0)_{\text{cn}}n^3 - x(0)^2n^2 - 2\dot{x}(0)y(0)n + \dot{x}(0)^2 + \dot{z}(0)^2 + 2y(0)^2n^2 + z(0)^2n^2\Big) = 0 \quad (8)$$

Note that if $x(0)_{\text{cn}}$ is set to zero, then the condition for drift-free second-order terms is

$$\dot{y}(0)_{\text{cn}} = \frac{1}{6n^3}\Big(x(0)^2n^2 + 2\dot{x}(0)y(0)n - \dot{x}(0)^2 - \dot{z}(0)^2 - 2y(0)^2n^2 - z(0)^2n^2\Big) \quad (9)$$

The secular term cancellation now depends on all five initial conditions from the Hill's solution in Eq. 2. The value of $\dot{y}(0)_{\text{cn}}$ is orders of magnitude larger than the Hill's initial conditions. This is reduced by $\epsilon \ll 1$ when recombined with the state (see Eq. 5).

Ref. [16] discusses techniques for combining the nonlinearity state vector with a state vector that is propagated using a model that includes the reference orbit eccentricity. The result is a solution that accounts for both eccentricity and second order nonlinearity effects,

$$X(t) = X_{\text{le}}(t) + \epsilon X_{\text{cn}}(t) \quad (10)$$

where $X_{\text{le}}(t)$ are the relative states given by a linear eccentric propagator (such as the one developed in subsection) and $X_{\text{cn}}(t)$ are given by Eqs. 15–20. The new initial conditions for a formation in the Hill's frame are

$$\begin{aligned}X(0) &= X_{\text{le}}(0) + \epsilon X_{\text{cn}}(0)\\ &= X_{\text{le}}(0) + \epsilon\begin{bmatrix} 0 & 0 & 0 & 0 & \dot{y}(0)_{\text{cn}} & 0 \end{bmatrix}^{\text{T}} \quad (11)\end{aligned}$$

Note that Tillerson presents several approaches for finding optimal initial conditions for $X_{\text{le}}(0)$ for a

$$
\begin{aligned}
\frac{y_{\mathrm{h}}^2+z_{\mathrm{h}}^2-2x_{\mathrm{h}}^2}{2} &= \frac{1}{2}\left(-2\frac{\dot{x}(0)}{n_{\mathrm{ref}}}+2\frac{\dot{x}(0)\cos(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}+y(0)-2x(0)\sin(n_{\mathrm{ref}}t)\right)^2 \\
&\quad +\frac{1}{2}\left(\frac{\dot{z}(0)\sin(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}+z(0)\cos(n_{\mathrm{ref}}t)\right)^2-\left(\frac{\dot{x}(0)\sin(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}+x(0)\cos(n_{\mathrm{ref}}t)\right)^2 \\
x_{\mathrm{h}}y_{\mathrm{h}} &= \left(x(0)\cos(n_{\mathrm{ref}}t)+\frac{\dot{x}(0)\sin(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}\right)\left(-2\frac{\dot{x}(0)}{n_{\mathrm{ref}}}+2\frac{\dot{x}(0)\cos(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}+y(0)-2x(0)\sin(nt)\right) \\
x_{\mathrm{h}}z_{\mathrm{h}} &= \left(x(0)\cos(n_{\mathrm{ref}}t)+\frac{\dot{x}(0)\sin(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}\right)\left(z(0)\cos(n_{\mathrm{ref}}t)+\frac{\dot{z}(0)\sin(n_{\mathrm{ref}}t)}{n_{\mathrm{ref}}}\right) \qquad (6)
\end{aligned}
$$

given initial position.[5] The nonlinearity correction developed in this Section can be used to find initial conditions that substantially reduce secular drift for widely spaced formations, with the new capability to specify 5 of the 6 possible initial conditions. Additionally, the form in Eq. 10 can be used to propagate the relative orbits of the satellites of the formation. This approach has been used to create recurring tetrahedron formations with sides extending beyond 10 km, whereas previous approaches restricted formations to have inter-spacecraft separations of approximately 0.1 km.

Planner Development

The previous section presented a way to initialize spacecraft in drift free formations. In practice, there are always differential disturbances acting on the spacecraft in a formation that will result in drift, even when it has been initialized correctly. As a result, feedback control will be required to maintain the formation geometry. A linear programming (LP) trajectory planning approach has been developed to design fuel-optimized trajectories and station-keeping control inputs.[3] The basic form of the LP is

$$\min \|u\|_1 \quad \text{subject to} \quad \mathbf{A}u \le \mathbf{b} \qquad (12)$$

where u is the vector of fuel inputs (ΔV) at each time step and $\mathbf{A}, \mathbf{b}$ are functions of the linearized spacecraft dynamics, initial conditions, and final conditions. The LP determines the control inputs for a specified time interval that minimizes the fuel cost (the sum of the inputs) while satisfying the constraints on the trajectory. Constraints to the problem can include state constraints such as remaining within some tolerance of a specified point, maximum input values (actuator saturation), and terminal constraints. This approach can include differential disturbances such as drag and linearized forms of the differential J$_2$ effects.[3] To complete the low-level control design, the LP is also embedded within a real-time optimization control approach that monitors spacecraft relative positions and velocities, and then redesigns the control input sequence if the vehicle approaches the edge of the error box.[3]

A mission with a highly elliptical orbit (*e.g.,* MMS) will require a propagator that accounts for eccentricity. Two common approaches to propagating relative states in eccentric orbits are Lawden's equations[10] and Melton's equations.[11] Melton's approach is in the time-domain, but is only valid for eccentricities up to 0.3, which is much less than that required for MMS.[11] Lawden's equations are valid for all eccentricities, but are written as a function of the true anomaly. Tillerson presented a relatively simple strategy of designing the trajectories as a function of the true anomaly, and then converting back to the time-domain for implementation. However, that is a complex process to perform in real-time and can introduce errors if the commands are not implemented in the correct way. The following presents a variation on Lawden's equations based on a derivation presented by Inalhan[4] that corrects this problem.

The following is a linear time-varying (LTV) propagator that is valid for any elliptical orbit,

$$
\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix}_j = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ a_{41} & \ddot{\theta}_{\mathrm{ref}} & 0 & 0 & 2\dot{\theta}_{\mathrm{ref}} & 0 \\ -\ddot{\theta}_{\mathrm{ref}} & a_{52} & 0 & -2\dot{\theta}_{\mathrm{ref}} & 0 & 0 \\ 0 & 0 & a_{63} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}_j + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}_j \qquad (13)
$$

where,

$$
\begin{aligned}
a_{41} &= \dot{\theta_{\mathrm{ref}}}^2 + 2n_{\mathrm{ref}}^2\left(\frac{1+e\cos\theta_{\mathrm{ref}}}{1-e^2}\right)^3 \\
a_{52} &= \dot{\theta_{\mathrm{ref}}}^2 - n_{\mathrm{ref}}^2\left(\frac{1+e\cos\theta_{\mathrm{ref}}}{1-e^2}\right)^3
\end{aligned}
$$

$$a_{63} = -n_{\text{ref}}^2 \left(\frac{1 + e\cos\theta_{\text{ref}}}{1 - e^2} \right)^3$$

and the reference orbit parameters are θ_{ref} – true anomaly, n_{ref} – period, and e – eccentricity. Also, using Refs. [8, 9],

$$\dot{\theta}_{\text{ref}} = \frac{n_{\text{ref}}}{(1 - e^2)^{3/2}} \left[(e\cos\theta_{\text{ref}})^2 + 2e\cos\theta_{\text{ref}} + 1 \right]$$

$$\ddot{\theta}_{\text{ref}} = \frac{-2en_{\text{ref}}\dot{\theta}_{\text{ref}}\sin\theta_{\text{ref}}}{(1 - e^2)^{3/2}}(e\cos\theta_{\text{ref}} + 1)$$

This propagator is given as a function of the true anomaly, making it parameter varying. But using Kepler's equation, an accurate mapping between elapsed time and true anomaly of the reference orbit can easily be created.[8] If such a mapping is created before the planning step, then θ_{ref} will be a known function of time, and the equations can effectively be rewritten as being linear time-varying. This result is a simple way to propagate a system in a highly elliptical orbit using fixed time-steps. This enables the use of the time-varying discretized form of these dynamics with the LP optimization technique in Eq. 12 and thereby extends the range of applications where the planner developed in Ref.[3] can be used effectively. Furthermore, the need for a real-time domain conversion while executing the resulting plan is eliminated, instead shifting added computation to the pre-planning phase, before any optimization takes place.

Fault Detection and Recovery

The purpose of the fault detection system is to monitor the health and status of the satellite network, and to recover from network link or end-point (satellite) failure by adapting the network topology to compensate for the failed end-point or broken communication link in the network.

The basic design of the fault-detection system is similar to a "virtual" token-ring. The main components are the fault detection system onboard the satellite, a "ring" network connecting all of the fault detectors sequentially, and a token which is relayed around the ring from satellite to satellite via the fault detectors. A fault occurs in the network when one of the satellite communication links goes down:– when this occurs the "ring" will be broken. The fault detection system will detect this failure and adapt the ring topology to bypass the failed satellite. When referring to the "ring" topology in the fault-detection algorithm, it should be noted that the topology is "virtual" or "logical", with no physical significance (*i.e.,* the logical path taken by fault detection tokens around the network is independent of the actual physical network topology).

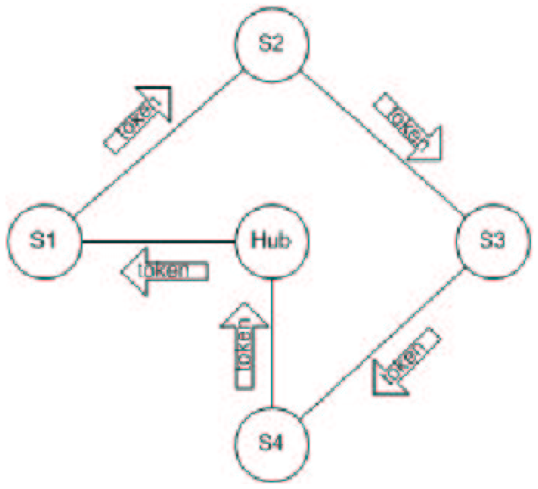

Fig. 3: Logical or "Virtual" Token Ring Topology

Figure 3 is the view of the network as seen by the fault detectors on board the satellites. The network is a logical token ring in which the token originates at the hub satellite; the hub sends it to satellite S_1, satellite S_1 in turn passes the token on to its successor S_2, and so on until the token arrives back at the hub. This simple ring topology is not the actual physical topology which the token traverses when going from satellite to satellite.

FORMATION FLYING DEMONSTRATIONS

Several demonstrations of the testbed's capabilities have been conducted.

LEO Small Aperture Demonstration

An initial test of ObjectAgent's capabilities is a simulation that included three Spacecraft agents and one Propagator agent. The spacecraft agents are each running an LQR controller, and stabilizing about individual locations about a reference ellipse (Hill's frame) generated off-line in MATLAB. Control commands are sent to the propagator agent, which propagates the spacecraft using Hill's equations. The propagator agent then incrementally sends position and velocity data to the spacecraft agents.

Although the effective sample rate (*i.e.,* the rate used by both the propagator and spacecraft agents) is fixed at 4 seconds, the simulation runs faster than real time, with an update rate limited only by the speed of the computers in the testbed. ObjectAgent showed no signs of instability during this testing phase.

Initial Conditions Evaluation

The initial condition correction for nonlinearity developed in Section (called NL) was tested against initializing with eccentricity corrections (EC), initializing with both eccentricity and nonlinearity corrections (NLEC), and initializing ignoring all corrections (NO). In each case, a tetrahedron with 10 km sides was created in an orbit with a period of 0.0824 days and an eccentricity of 0.05. The simulation was conducted with a fully nonlinear propagator, but without the presence of drag or J_2 disturbances. The quality factor Q_{GM}, was used to compare the approaches, where

$$Q_{GM} = \frac{\text{True Volume}}{\text{Ideal Volume}} + \frac{\text{True Surface}}{\text{Ideal Surface}} + 1 \quad (14)$$

where *True Volume* is the volume of the tetrahedron (in this case, formed by the four spacecraft), *Ideal Volume* is the volume of a regular tetrahedron based on the average side length of the tetrahedron, *True Surface* is the surface area of the tetrahedron, and *Ideal Surface* is the surface area of a regular tetrahedron based on the average side length of the tetrahedron.[13, 15] Q_{GM} ranges between 3.0 (a regular tetrahedron) and 1.0 (a line). There are a variety of commonly used tetrahedron quality factors, but Q_{GM} was used because it is consistently capable of unambiguously identifying regular tetrahedrons.[15] The tetrahedron shape is designed to appear only at apogee, so measurements of Q_{GM} are made once per orbit at whatever point the shape is most regular.

Results for the four initialization cases are shown in Figure 4, which plots the Q_{GM} trend after each orbit. The results clearly show that the EC and NLEC initializations maintain their shapes much longer than the NO and NL. It is also clear that correctly accounting for the orbit eccentricity has a significant influence on the quality of the tetrahedron, confirming the analysis in[4] (NL $\Rightarrow$ EC). The improvement from the eccentricity corrected to the combined nonlinearity/eccentricity corrected initialization (EC $\Rightarrow$ NLEC) is smaller but still clearly important for reducing the shape deformation over time.[16]

A second simulation was conducted with the same formation, but with a highly elliptical orbit (period of 1 day, $e = 0.82$). The results of that simulation are shown in Figure 5. The results in Figure 5 use $3 - Q_{GM}$ as a metric, because the tetrahedrons are very regular in this case. It can be seen that over 40 orbits, the quality of the tetrahedron in the formation decreases negligibly and at nearly the same rate for both the EC and NLEC initializations. Figure 6 compares the effects of using the EC and NLEC initializations at different separations in the same highly eccentric orbit. In each case, the NLEC initialization results in a decrease in tetrahedron quality. These results suggest that decoupling in Eq. 11 that is used in the initialization is invalid for large eccentricities, but further investigation is required.

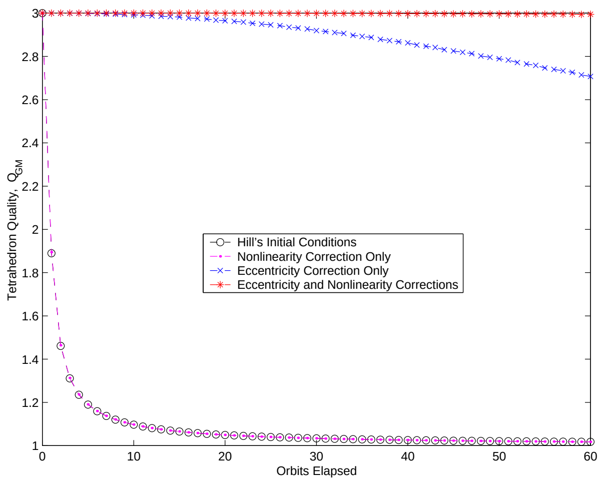

Fig. 4: Effect of Initial Conditions on Tetrahedron Quality in Low Earth Orbit ($e = 0.05$)

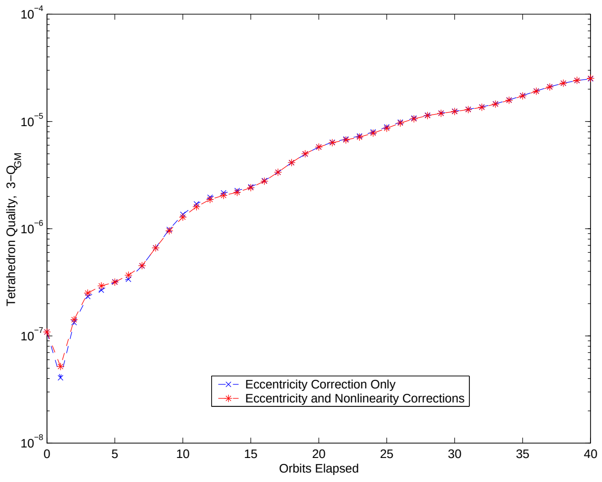

Fig. 5: Effect of Initial Conditions on Tetrahedron Quality in a Highly Eccentric Orbit ($e = 0.82$)

Elliptical Tetrahedron Demonstration

The testbed was used to conduct a demonstration of a mission similar to one segment of the MMS mission. Four spacecraft were arranged in a tetrahedron formation, where the sides of the tetrahedron were each 10 km. The propagator used in the simulation agent was a Runge-Kutta 7-8 routine that accounts for both J_2 and drag for the osculating orbit. The linear time-varying propagator developed earlier was

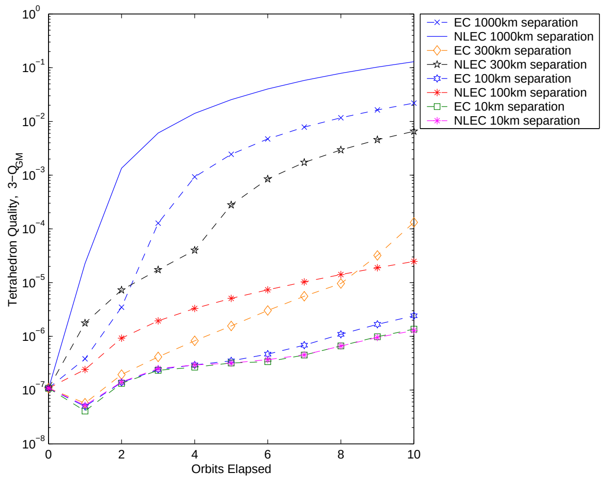

Fig. 6: The Effect of Initial Conditions and Inter-spacecraft Separation on Tetrahedron Quality in a Highly Eccentric Orbit

used for the relative orbits. The initial conditions of the satellites were chosen to create drift-free relative orbits that form a tetrahedron configuration once per day (for an osculating orbit with a period of 24 hours). In this demonstration, the simulation engine and four spacecraft are being run from five computers at Cornell and MIT.

The initial configuration of the formation is shown in Figure 7 and Figures 8 and 9 show the simulation evolving over time. The plots show that the configuration reforms a tetrahedron after a full orbit. Lines have been drawn to emphasize tetrahedron geometry when it is present. The initial conditions of the osculating orbit were chosen to match the highly elliptical orbit used in Ref. [12]. The relative initial conditions of each spacecraft were chosen to lie on a tetrahedron, thereby fixing the initial relative positions. Initial relative velocities were chosen according to the optimization algorithm first presented in Ref. [5].

While simulating the spacecraft and the dynamics, the testbed tracks the communication used by each agent each spacecraft. Figure 10 shows a plot of communication usage by each satellite after a full orbit. The lower half of the graph shows the instantaneous number of bytes being sent ($*$) and received ($+$) by each satellite in the testbed. The upper half of the graph shows the total number of bytes being sent by all agents over the history of the simulation. The graph shows fewer bytes being sent than received, because messages with multiple destinations have only been counted multiple times on receipt.

During the demonstration, a catastrophic failure

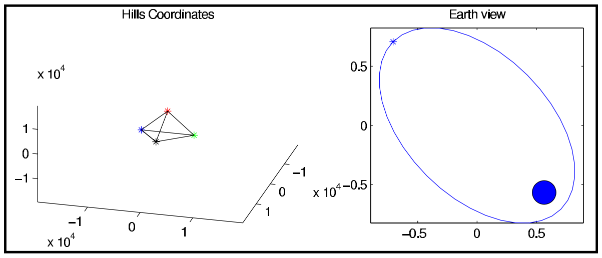

Fig. 7: Initial Configuration

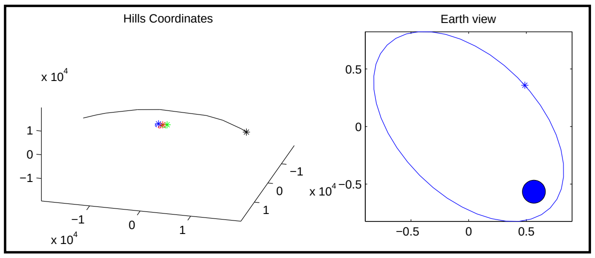

Fig. 8: State after partial orbit

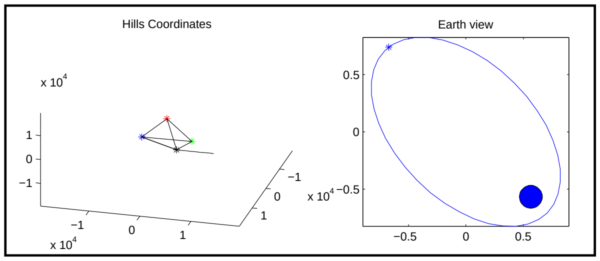

Fig. 9: State after full orbit

was inserted into one of the spacecraft. The failure mechanism was the removal of that spacecraft's computer from the testbed network. The result was an immediate drop in overall communications and complete loss of instantaneous communication for the failed satellite. This demonstration shows the testbed's versatility as a tool for studying different relevant mission classes from a variety of different perspectives. The framework provided by OA proved to be particularly well suited to the development of the simulation and to the monitoring of interprocess and network communications.

CONCLUSIONS

New technologies have been developed to support emerging formation flying missions. An initialization scheme for creating widely-separated formations was extended to a more general case that allows for any three-dimensional drift-free geometry (*i.e.*, a tetrahedron). This approach works well for low eccentricity orbits, but is less useful for the highly eccentric orbits planned for the MMS mission. A new linear relative orbit propagator was also presented, which is useful for real-time planning for missions in highly eccentric orbits. It is valid for any eccentric-

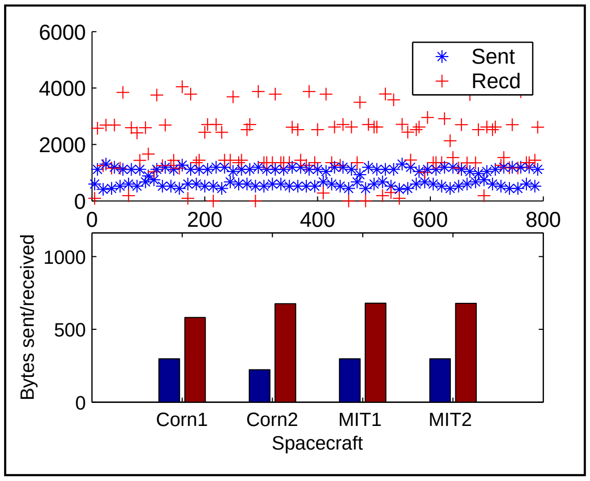

Fig. 10: Communications after full orbit

ity, but it does require a mapping between the times considered for thruster firings and the true anomaly of the reference orbit at those times. The primary advantage of this approach is that it shifts this conversion to the pre-planning phase, before any optimization takes place. This should reduce the real-time processor load.

A functional spacecraft formation-flying testbed has been created using the OA middleware. The testbed was used in both single and multiple-computer configurations to demonstrate the initialization of tetrahedron formations in various orbits. Using OA significantly reduced the time for the development of a highly distributed testbed because network communication functions were abstracted and rigid interfaces between software components were enforced. This testbed will also be used to study the interplay between inter-spacecraft communication, autonomy, and formation control requirements. The testbed's emphasis on communications and its ability to simulate formation flying spacecraft by placing each spacecraft on its own computer running only compiled "flight software" in variably real-time environments, makes it uniquely suited to a realistic evaluation of these technologies.

ACKNOWLEDGMENTS

Research funded under NASA Grant #NAG5-10440.

REFERENCES

[1] J. Leitner, F. Bauer, D. Folta, M. Moreau, R. Carpenter, J. How, "Distributed Spacecraft Systems Develop New GPS Capabilities," in *GPS World: Formation Flight in Space* Feb. 2002.

[2] D.M. Surka, M.C. Brito, and C.G. Harvey, "Development of the Real-Time Object-Agent Flight Software Architecture for Distributed Satellite Systems," *IEEE Aerospace Conf.,* 2001.

[3] M. Tillerson, G. Inalhan, and J. How, "Coordination and Control of Distributed Spacecraft Systems Using Convex Optimization Techniques," *International Journal of Robust and Nonlinear Control,* vol 12, Issue 2-3, Feb.-Mar. 2002, p.207-242.

[4] G. Inalhan, M. Tillerson, J. How, "Relative Dynamics & Control of Spacecraft Formations in Eccentric Orbits," AIAA *JGCD* vol. 25, no. 1, Jan.-Feb. 2002, p. 48-59.

[5] M. Tillerson, J. How, "Formation Flying Control in Eccentric Orbits," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Montreal, Canada, Aug. 6-9, 2001,.

[6] P. Ferguson, C. Park, M. Tillerson, and How, J. P., "New Formation Flying Testbed for Analyzing Distributed Estimation and Control Architectures," *AIAA GN&C Conference,* Aug 2002.

[7] P. Ferguson, J. How, "Decentralized Estimation Algorithms for Formation Flying Spacecraft". To appear at the *AIAA Guidance, Navigation and Control Conference*, Austin, Texas, August 2003.

[8] D. A. Vallado. *Fundamentals of Astrodynamics and Applications.* McGraw-Hill, 1997.

[9] J. P. Vinti. *Orbital and Celestial Mechanics.* AIAA, 1998, pp.196-197.

[10] D. F. Lawden, *Optimal Trajectories for Space Navigation,* Butterworths, London, 1963.

[11] R. G. Melton, Time Explicit Representation of Relative Motion Between Elliptical Orbits, *JGCD* Vol. 23, No. 4, July - Aug. 2000, pp. 604-610.

[12] L. Mailhe, C. Schiff, and S. Hughes, "Formation Flying in Highly Elliptical Orbits: Initializing the Formation," *Proceedings of the International Symposium on Space Dynamics,* Biarritz, France, CNES, June 26-30 2000.

[13] J. Guzman and C. Schiff, "A Preliminary Study for a Tetrahedron Formation (Spacecraft Formation Flying)," *AIAA/AAS Astro. Specialists Conf.*, Aug 2002.

[14] S. Curtis, "The Magnetospheric Multiscale Mission Resolving Fundamental Processes in Space Plasmas," NASA GSFC, Greenbelt, MD, Dec. 1999. NASA/TM2000-209883.

[15] P. Robert, A. Roux, C. Harvey, M. Dunlop, P. Daly, and K. Glassmeier, "Tetrahedron Geometric Factors," Analysis Methods for Multi-Spacecraft Data (G. Paschmann and P. Daly, eds.), pp. 323348, Noordwijk, The Netherlands: ISSI Report SR-001, ESA Pub. Div., 1998.

[16] K. T. Alfriend, H. Schaub, and D.-W. Gim, "Formation Flying: Accomodating Non-linearity and Eccentricity Perturbations," presented at the 12th *AAS/AIAA Space Flight Mechanics Meeting,* January 27-30, 2002.

[17] K. T. Alfriend, H. Schaub, and D.-W. Gim, "Gravitational Perturbations, Nonlinearity and Circular Orbit Assumption Effects on Formation Flying Control Strategies," Vol. 104, *Advances in the Astronautical Sciences, Guidance and Control* p. 139.

[18] C. W. Park, *Precise Relative Navigation using Augmented CDGPS.* Ph.D. Dissertation, Stanford University, Dept. of Mechanical Eng., June 2001.

[19] F. D. Busse, *Precise Formation-State Estimation in Low Earth Orbit using Carrier Differential GPS.* Ph.D. Dissertation, Stanford University, Dept. of Aeronautics and Astronautics, November 2002.

APPENDIX

The following equations describe the time evolution of a second-order nonlinearity correction state for a spacecraft being propagated in a Hill's frame. They are the solutions to the integrals in Eqs. 15–20.

$$
\begin{aligned}
x_{cn}(t) \;=\; & \frac{1}{12n^4}\Bigg(48x(0)_{\rm cn}n^4 - 36x(0)_{\rm cn}n^4\cos(nt) + 12\sin(nt)vx(0)_{\rm cn}n^3 + 24\dot{y}(0)_{\rm cn}n^3 - 24\dot{y}(0)_{\rm cn}n^3\cos(nt) \\
& + 3z(0)^2n^2 - 6x(0)^2n^2 + 4x(0)\dot{x}(0)\sin(2nt)n + \cos(2nt)\dot{z}(0)^2 - 2\cos(2nt)\dot{x}(0)^2 + 6y(0)^2n^2 + 8\dot{x}(0)^2\cos(nt) \\
& - \cos(2nt)z(0)^2n^2 + 2\cos(2nt)x(0)^2n^2 - 2\sin(2nt)n\dot{z}(0)z(0) - 8x(0)\sin(nt)n\dot{x}(0) - 4\cos(nt)\dot{z}(0)^2 \\
& + 4n\sin(nt)\dot{z}(0)z(0) + 4\cos(nt)x(0)^2n^2 - 6\cos(nt)y(0)^2n^2 - 2\cos(nt)z(0)^2n^2 + 3\dot{z}(0)^2 - 6\dot{x}(0)^2\Bigg)
\end{aligned} \tag{15}
$$

$$
\begin{aligned}
y_{cn}(t) \;=\; & x(0)_{\rm cn}[-6nt + 6\sin(nt)] + y(0)_{\rm cn} + \frac{\dot{x}(0)_{\rm cn}}{n}[-2 + 2\cos(nt)] + \frac{\dot{y}(0)_{\rm cn}}{n}[4\sin(nt) - 3nt] - \frac{1}{n^4}\Bigg(\frac{2}{3}\sin(nt)x(0)^2n^2 \\
& - \frac{1}{3}\sin(nt)z(0)^2n^2 - \sin(nt)y(0)^2n^2 - x(0)n^2y(0) + \frac{1}{2}x(0)n\dot{x}(0) + \frac{1}{2}n\dot{z}(0)z(0) - \frac{1}{12}\sin(2nt)x(0)^2n^2 \\
& - \frac{1}{12}\sin(2nt)z(0)^2n^2 - \frac{2}{3}\dot{x}(0)^2\sin(nt) - \frac{2}{3}\sin(nt)\dot{z}(0)^2 - \frac{2}{3}n\cos(nt)\dot{z}(0)z(0) + x(0)\cos(nt)n^2y(0) \\
& - \frac{2}{3}x(0)\cos(nt)n\dot{x}(0) + \dot{x}(0)\sin(nt)y(0)n + \frac{1}{6}\cos(2nt)nx(0)\dot{x}(0) + \frac{1}{6}\cos(2nt)n\dot{z}(0)z(0) - \frac{1}{2}n^3tx(0)^2 \\
& - t\dot{x}(0)y(0)n^2 + \frac{1}{2}t\dot{x}(0)^2n + \frac{1}{2}nt\dot{z}(0)^2 + n^3ty(0)^2 + \frac{1}{2}n^3tz(0)^2 + \frac{1}{12}\sin(2nt)\dot{x}(0)^2 + \frac{1}{12}\sin(2nt)\dot{z}(0)^2\Bigg)
\end{aligned} \tag{16}
$$

$$
\begin{aligned}
z_{cn}(t) \;=\; & z(0)_{\rm cn}\cos(nt) + \frac{\dot{z}(0)_{\rm cn}}{n}\sin(nt) + \frac{1}{n^4}\Bigg(\frac{1}{2}x(0)z(0)n^2 - \frac{1}{6}\sin(2nt)nx(0)\dot{z}(0) - \frac{1}{6}\sin(2nt)n\dot{x}(0)z(0) \\
& + \frac{1}{2}\dot{x}(0)\dot{z}(0) - \frac{1}{6}\cos(2nt)x(0)z(0)n^2 + \frac{1}{6}\cos(2nt)\dot{x}(0)\dot{z}(0) + \frac{1}{3}x(0)\sin(nt)n\dot{z}(0) + \frac{1}{3}n\sin(nt)\dot{x}(0)z(0) \\
& - \frac{1}{3}x(0)\cos(nt)z(0)n^2 - \frac{2}{3}\cos(nt)\dot{x}(0)\dot{z}(0)\Bigg)
\end{aligned} \tag{17}
$$

$$
\begin{aligned}
\dot{x}_{cn}(t) \;=\; & 3n\sin(nt)x(0)_{\rm cn} + \cos(nt)\dot{x}(0)_{\rm cn} + 2\sin(nt)\dot{y}(0)_{\rm cn} + \frac{1}{3n^3}\Bigg(2\cos(2nt)nx(0)\dot{x}(0) - 2x(0)\cos(nt)n\dot{x}(0) \\
& - \cos(2nt)n\dot{z}(0)z(0) + \frac{1}{2}\sin(2nt)z(0)^2n^2 - \sin(2nt)x(0)^2n^2 - \frac{1}{2}\sin(2nt)\dot{z}(0)^2 + \sin(2nt)\dot{x}(0)^2 - 2\dot{x}(0)^2\sin(nt) \\
& + n\cos(nt)\dot{z}(0)z(0) + \frac{1}{2}\sin(nt)z(0)^2n^2 - \sin(nt)x(0)^2n^2 + \frac{3}{2}\sin(nt)y(0)^2n^2 + \sin(nt)\dot{z}(0)^2\Bigg)
\end{aligned} \tag{18}
$$

$$
\begin{aligned}
\dot{y}_{cn}(t) \;=\; & (-6n + 6n\cos(nt))x(0)_{\rm cn} - 2\sin(nt)\dot{x}(0)_{\rm cn} + \frac{1}{n^3}\Bigg(-3 + 4\cos(nt))\dot{y}(0)_{\rm cn} + (-\frac{1}{2}z(0)^2n^2 + \frac{1}{2}x(0)^2n^2 \\
& - y(0)^2n^2 + \dot{x}(0)y(0)n - \frac{1}{2}\dot{z}(0)^2 - \frac{1}{2}\dot{x}(0)^2 - \cos(nt)\dot{x}(0)y(0)n \\
& + \frac{1}{3}\sin(2nt)n\dot{z}(0)z(0) + x(0)\sin(nt)n^2y(0) - \frac{2}{3}x(0)\sin(nt)n\dot{x}(0) + \frac{1}{3}x(0)\dot{x}(0)\sin(2nt)n + \frac{2}{3}\dot{x}(0)^2\cos(nt) \\
& - \frac{1}{6}\cos(2nt)\dot{z}(0)^2 - \frac{1}{6}\cos(2nt)\dot{x}(0)^2 + \frac{1}{6}\cos(2nt)z(0)^2n^2 + \frac{1}{6}\cos(2nt)x(0)^2n^2 - \frac{2}{3}n\sin(nt)\dot{z}(0)z(0) \\
& - \frac{2}{3}\cos(nt)x(0)^2n^2 + \cos(nt)y(0)^2n^2 + \frac{1}{3}\cos(nt)z(0)^2n^2 + \frac{2}{3}\cos(nt)\dot{z}(0)^2\Bigg)
\end{aligned} \tag{19}
$$

$$
\begin{aligned}
\dot{z}_{cn}(t) \;=\; & -n\sin(nt)z(0)_{\rm cn} + \cos(nt)\dot{z}(0)_{\rm cn} - \frac{1}{n^3}\Bigg(-\frac{1}{3}\sin(2nt)x(0)z(0)n^2 + \frac{1}{3}\sin(2nt)\dot{x}(0)\dot{z}(0) \\
& + \frac{1}{3}\cos(2nt)nx(0)\dot{z}(0) + \frac{1}{3}\cos(2nt)n\dot{x}(0)z(0) - \frac{1}{3}\sin(nt)x(0)z(0)n^2 - \frac{2}{3}\sin(nt)\dot{x}(0)\dot{z}(0) \\
& - \frac{1}{3}n\cos(nt)x(0)\dot{z}(0) - \frac{1}{3}n\cos(nt)\dot{x}(0)z(0)\Bigg)
\end{aligned} \tag{20}
$$