Decentralized Formation Flying Control in a Multiple-Team Hierarchy

Joseph B. Mueller[*]

Stephanie J. Thomas[†]

Princeton Satellite Systems, Princeton, NJ 08542

In recent years, formation flying has been recognized as an enabling technology for a variety of mission concepts in both the scientific and defense arenas. Examples of developing missions at NASA include MMS (Magnetospheric Multi-Scale), SIRA (Solar Imaging Radio Array), and TPF (Terrestrial Planet Finder). For each of these missions, a multiple-satellite approach is required in order to accomplish the large-scale geometries imposed by the science objectives. In addition, the paradigm shift of using a multiple-satellite cluster rather than a large, monolithic spacecraft has also been motivated by the expected benefits of increased robustness, greater flexibility, and reduced cost. However, the operational costs of monitoring and commanding a fleet of close-orbiting satellites is likely to be unreasonable unless the onboard software is sufficiently autonomous, robust, and scalable to large clusters.

This paper presents the prototype of a system that addresses these objectives – a decentralized guidance and control system that is distributed across spacecraft using a multiple-team framework. The objective is to divide large clusters into teams of "manageable" size, so that the communication and computational demands driven by N decentralized units are related to the number of satellites in a team rather than the entire cluster. The system is designed to provide a high-level of autonomy, to support clusters with large numbers of satellites, to enable the number of spacecraft in the cluster to change post-launch, and to provide for on-orbit software modification.

The distributed guidance and control system will be implemented in an object-oriented style using MANTA (Messaging Architecture for Networking and Threaded Applications). In this architecture, tasks may be remotely added, removed or replaced post-launch to increase mission flexibility and robustness. This built-in adaptability will allow software modifications to be made on-orbit in a robust manner. The prototype system, which is implemented in MATLAB, emulates the object-oriented and message-passing features of the MANTA software.

In this paper, the multiple-team organization of the cluster is described, and the modular software architecture is presented. The relative dynamics in eccentric reference orbits is reviewed, and families of periodic, relative trajectories are identified, expressed as sets of static geometric parameters. The guidance law design is presented, and an example reconfiguration scenario is used to illustrate the distributed process of assigning geometric goals to the cluster. Next, a decentralized maneuver planning approach is presented that utilizes linear-programming methods to enact reconfiguration and coarse formation keeping maneuvers. Finally, a method for performing online collision avoidance is discussed, and an example is provided to gauge its performance.

---

[*]Senior Technical Staff, 33 Witherspoon St., jmueller@psatellite.com, (763) 560-9613

[†]Senior Technical Staff, 33 Witherspoon St., sjthomas@psatellite.com, (609) 279-9606

1. Introduction

The ability to provide sustained control over the relative motion of close-orbiting satellites is a requirement for several upcoming missions in both the scientific and defense arenas. Examples of currently planned formation flying missions at NASA include MMS (Magnetospheric Multi-Scale), SIRA (Solar Imaging Radio Array), and TPF (Terrestrial Planet Finder). For each of these missions, a multiple-satellite approach is required in order to accomplish the large-scale geometries imposed by the science objectives. In addition, though, the growing interest in formation flying technology has been fueled by several supposedly inherent advantages.

A distributed system of spacecraft is expected to be more robust to failures and to provide greater flexibility than a single spacecraft. Robustness is improved if the cluster can continue to function in spite of a failure or degradation that may occur in a subset of spacecraft. Flexibility is enhanced if the formation can change its geometry during the mission in response to changing science events or mission objectives. For relatively small clusters, these capabilities may be realized through the traditional ground-based operation of satellites. However, as the number of satellites is increased, the cost and complexity of managing a large fleet of close-orbiting spacecraft individually from the ground is likely to become unreasonable. In order to exploit the advantages inherent in a distributed system, and to facilitate a feasible level of ground-based control, the onboard software must be sufficiently autonomous, robust, and scalable to large clusters.

This paper presents the prototype of a decentralized formation flying (DFF) control system that addresses the issues of autonomy, robustness, and scalability. In particular, the system is designed to:

1. accommodate a wide range of orbits and formation geometries;

2. support different types of spacecraft configurations;

3. provide for a substantial degree of autonomous operation, including:

    - automatic formation initialization;
    - planning and execution of reconfiguration maneuvers; and
    - preemptive collision avoidance

4. be scalable to large clusters, through the use of a multiple-team organization; and

5. allow software modifications to be made safely and efficiently throughout the mission.

The prototype DFF system is being developed into a distributed software application that can be adapted with minimal modification to suit a variety of missions. The system is being designed in an object-oriented style using MANTA middleware,[1] which is written in C++. MANTA stands for "Messaging Architecture for Networking and Threaded Applications". In this architecture, tasks may be remotely added, removed or replaced at run-time to increase mission flexibility and robustness. In addition, each task offers a user-configurable telemetry stream, so that the internal states and actions of the software are highly visible to the ground operator. This built-in adaptability and visibility will allow software modifications to be made on-orbit in a safe manner. The prototype system, which is implemented in MATLAB, emulates the object-oriented and message-passing features of the MANTA software.

This paper first describes the scope and architectural layout of the DFF software, and then explains the motivation for using a multiple-team organization. The remainder of the paper focuses on the algorithms that drive the system. The relative dynamics in eccentric reference orbits is summarized, and families of periodic, relative trajectories are identified, expressed as sets of static geometric parameters. The guidance law design is presented, and an example reconfiguration scenario is used to illustrate the distributed process of assigning geometric goals to the cluster. Next, a decentralized maneuver planning approach is presented that utilizes linear-programming methods to enact reconfiguration and coarse formation keeping maneuvers. Finally, a method for performing online collision avoidance is discussed, and an example is provided to gauge its performance.

2. DFF System Overview

2.1. Scope

The scope of the DFF system is not intended to encapsulate every aspect of a formation flying control system. It is limited to the elements of the system that are essentially independent of the orbit and spacecraft configuration. In general, a complete system will consist of the following elements:

- Relative navigation
- **Guidance**
- **Initialization**
- **Reconfiguration**
- **Coarse formation maintenance**
- Precision formation control
- **Preemptive Collision avoidance**
- Reactionary collision avoidance
- Fault management

The bold items are provided by the DFF system – guidance, initialization, reconfiguration, coarse formation maintenance, and preemptive collision avoidance. These aspects of formation flying can be generalized to accommodate a variety of different orbits and spacecraft designs, while the other aspects are unique to every mission.

2.2. Software Architecture

A high-level block diagram of the DFF system is shown in Figure 1 on the following page. The dashed line surrounds the core DFF software, which is considered mission-independent. Each of the external software and hardware components require a separate interface. These interfaces are specific to the spacecraft design and are therefore implemented as "Interface plugins", separate from the core DFF system. The same software resides on every spacecraft in the cluster.

The system consists of 10 software modules. In the MANTA environment, the modules are implemented as separate, single-threaded tasks that communicate with one another through an efficient messaging system. In the MATLAB prototype, the modules are written as separate functions with persistent memory. The message-passing functionality and object-oriented design is emulated in MATLAB to facilitate a more direct transition to C++. The arrows connecting the modules indicate the flow of messages within the system. Individual connections are not shown for the Command Processing and Parameter Database, as they communicate with all other modules. The Team Management, Guidance Law, and Control Law modules require communication with other spacecraft. Inter-spacecraft communication is handled with an ISL Management module, which is designed to enable fault-tolerant message-passing throughout the cluster.

The Parameter Database module serves as a central repository for various types of data required by the system. It is initialized first from a set of scripts that define the mission and spacecraft parameters. It is then used to initialize all other modules, and is used throughout the mission for parameter changes and telemetry requests initiated by the ground station.

The relative navigation system is expected to provide the estimates of the absolute and relative states in an inertial cartesian frame centered at the team reference. The Coordinate Transformation module transforms the absolute position and velocity into mean orbital elements,[2] and transforms the relative state vector from the inertial frame into two different coordinate sets: Hill's-frame coordinates and orbital element differences (see Section 4 on page 6). This new relative state information is then provided to the other modules.

The main functionality of the DFF system is provided by the Team Management, Guidance Law and Control Law modules. The Team Management module facilitates the multiple-team organization of the cluster. It provides autonomous team formation capability, and maintains the information that defines the hierarchical team framework. Satellites may be added to or removed from teams dynamically, and the roles of reference and supervisor may be changed throughout the mission. The role of the Guidance Law is to define the desired relative trajectory for each spacecraft to follow, based upon the objectives supplied by the ground operator. It performs trajectory assignment during reconfiguration maneuvers, and determines
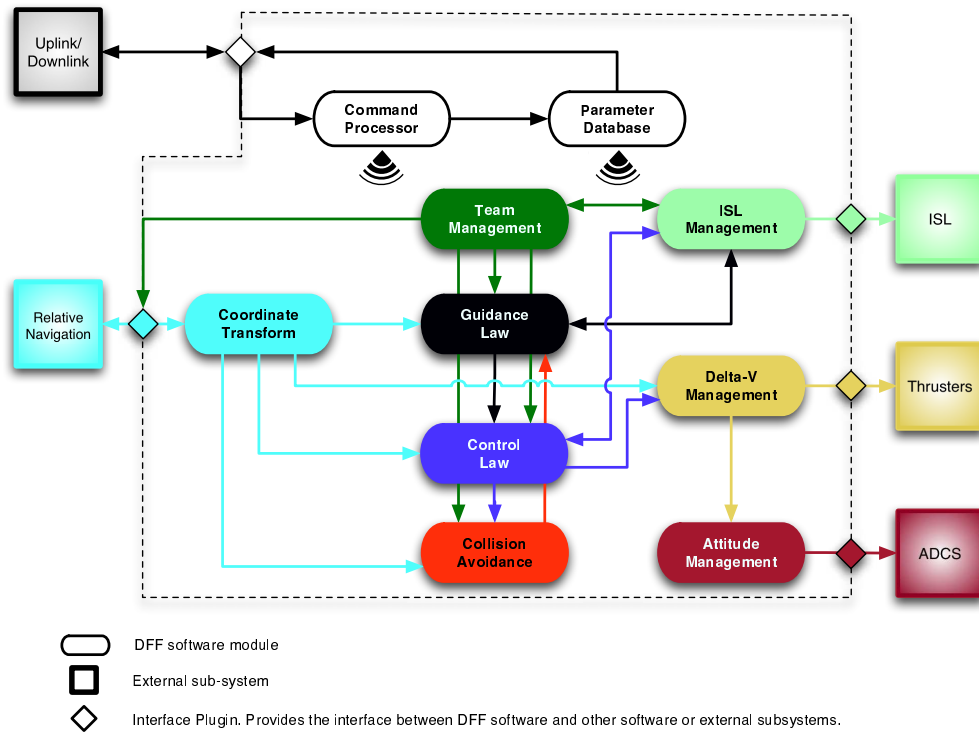
Figure 1. Block Diagram of the Task-Based Software Architecture for the DFF System

safe periodic trajectories during formation initialization. Once a desired trajectory has been defined by the Guidance Law, the Control Law is then responsible for planning maneuvers to achieve and maintain it.

The output of the Control Law module is a time-tagged sequence of delta-v's. When planning reconfiguration maneuvers, the Control Law distributes this data to the Collision Avoidance module on every spacecraft, so that the entire maneuver may be verified for safety before it is begun. The delta-v sequence is then supplied to the Delta-V Management module, which is responsible for commanding thruster valves to open at the specified time for the proper duration, so that the desired delta-v is attained. Prior to each thruster firing, this module also sends the desired orientation of the spacecraft (as a Hill's-frame to body-frame quaternion) to the Attitude Management module, so that the selected thruster(s) will be properly aligned during the burn. The Attitude Management module is then responsible for commanding the desired orientation to the attitude control system. For a spin-stabilized spacecraft, the right ascension and declination are commanded, while an ECI-to-body quaternion is commanded for a 3-axis stabilized configuration.

The guidance and control functionality is suitable for a wide range of spacecraft designs, and supports all possible types of naturally repeating formations in central body orbits. The algorithms have been developed for both circular and eccentric reference orbits. The details of the guidance and control algorithms are discussed in later sections. First, however, a description of the relative dynamics is offered.

3. Decentralization Using a Multiple-Team Framework

The design of a formation flying GN&C system is driven by several competing objectives. We simultaneously seek to reduce the level of inter-spacecraft communication, distribute the computational effort across satellites, and maintain complete control over the cluster. These objectives are the result of physical constraints associated with the navigation sensors, communication bandwidth, and processing power. The problem leads to an architectural decision for the control system, based upon the trade-offs between a *centralized* and a *decentralized* approach.

In a purely centralized system, one satellite serves as a hub or a reference, while all other satellites control their trajectory relative to it. This places greater demands on the relative navigation hardware and software

of the reference satellite. Since this satellite has complete knowledge of the cluster, it is also burdened with responsibility of collision avoidance, guidance and control. The advantage of this approach is that it provides global control over the cluster, and requires a minimum level of inter-spacecraft communication. The disadvantage is that it is prone to a single-point failure.

In a purely decentralized system, each satellite has awareness of all other satellites in the cluster. In this sense, a global relative state is shared throughout, and control is applied independently at each node. It has been shown that the data transmission requirements can be minimized with a decentralized LQG control approach.[3] Relative navigation can be decentralized using different types of Extended Kalman Filters.[4,5] The advantages of the decentralized architecture are that it distributes the computational load equally throughout the cluster and is robust to single-point failures. This approach is preferred as it utilizes the inherent robustness advantages associated with a distributed system.

As the number of satellites in the cluster grows, the demands on processing power and communication bandwidth increase rapidly. This is true for either architecture. It therefore becomes increasingly difficult to realize a purely centralized or purely decentralized system. A solution to this problem is to divide the cluster into smaller teams of manageable size. The cluster is decentralized, in that each team is controlled independently. Global control over the cluster is still maintained, however, as the teams are connected in a hierarchical framework. At the team level, a decentralized navigation and control architecture may be implemented. Figure 2 illustrates an example organization where 16 satellites are divided into 4 teams. This diagram was created automatically from a MATLAB utility which operates on an array of team data structures.
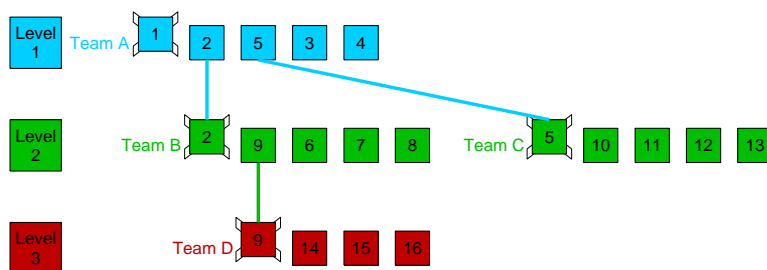

Figure 2. Example of a Multiple-Team Framework

Each satellite is shown with a unique ID. Every team consists of a *reference* and one or more *relatives*, where the reference defines the origin of that team's relative frame. The reference of each team is shown to the left of the relatives and is marked with an "X". All four teams are connected in a hierarchical fashion, with a relative on one team also serving as the reference to another. For example, Team A is connected to Team B in that one of the Team A relatives (5) serves as the reference for Team B. At the top of the hierarchy, the reference of Team A (1) is considered the *cluster reference*. This hierarchical organization corresponds only to the definition of the relative frames. It enables us to define a unique relative frame for each team, while maintaining a global reference frame for the entire cluster.

Within a given team, the relative satellites control their trajectories with respect to the reference. The maneuver planning for each member is conducted locally, so that the computational effort is distributed throughout the team. A significant portion of the guidance law is distributed, as well, with each member estimating the costs to achieve all possible target states. This cost data from each member is returned to the team *captain*, who applies an assignment algorithm to determine the optimal configuration. At any given time, one spacecraft in the team serves the role of captain, and is responsible for carrying out any tasks that require coordination. The guidance law is discussed in detail in Section 6 on page 11.

The primary reason for using a multiple-team framework is to alleviate the communication and processing demands associated with large clusters. In addition, it could be used to reduce the linearization errors associated with formations that have extremely large separation distances. However, dividing the cluster into teams has significant drawbacks. It will degrade the performance of the relative navigation system, tend to increase the total fuel usage of the cluster, and create an additional layer of complexity in the software. Therefore, the multiple-team approach should only be applied out of necessity – when the number of spacecraft or the size of the formation is so large that a globally decentralized approach is not feasible.

4.   Dynamics of Relative Motion

The equations governing the relative motion of close-orbiting satellites are available in a number of different forms. The well known Clohessy-Wiltshire equations[6] provide a linear time-invariant solution for the relative motion of one orbit with respect to a circular reference orbit. Lawden's equations[7] express the relative motion in eccentric orbits, with true anomaly as the independent variable. In recent years, Lawden's equations have been used to generate periodic relative trajectories in eccentric orbits.[8] In addition, recent work by Broucke[9] has provided a nonsingular solution to elliptic relative motion with time as the independent variable.

Relative motion is defined in a rotating reference frame, with the origin fixed to a reference point that orbits with the cluster. In this paper, the reference is always defined as one of the spacecraft. Figure 3 shows the relative frame, hereafter referred to as Hill's frame. The x-axis points in the zenith direction, the z-axis is aligned with the angular momentum vector, normal to the orbital plane, and the y-axis completes the right-hand system. For circular orbits, y is always aligned with the velocity vector.

Gauss' variational equations (GVEs) can also be used to express the relative dynamics in terms of differential orbital elements. It will be pointed out in Section 7 on page 14 that this coordinate frame provides a more accurate and computationally efficient method of predicting the relative motion. When designing relative trajectories for a formation, however, we are interested in the evolution of the rectilinear coordinates. For the purpose of formation design and initialization, it is therefore necessary to express the relative dynamics in Hill's-frame coordinates.

Figure 3.   Relative Frame

The LTV dynamic equations of a satellite's relative motion in Hill's frame are given as follows:

$$\frac{d}{dt}\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \dot{\nu}^2 + 2g & \ddot{\nu} & 0 & 0 & 2\dot{\nu} & 0 \\ -\ddot{\nu} & \dot{\nu}^2 - g & 0 & 0 & -2\dot{\nu} & 0 \\ 0 & 0 & -g & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \tag{1}$$

where ν is the true anomaly, e is the eccentricity, n is the mean orbit rate, u is the applied acceleration, and the g term is:

$$g = n^2 \left(\frac{1 + e\cos\nu}{1 - e^2} \right) \tag{2}$$

In addition, the first and second derivatives of ν are:

$$\dot{\nu} = \frac{n\,(1 + e\cos\nu)^2}{(1 - e^2)^{3/2}} \tag{3}$$

$$\ddot{\nu} = -(2e\dot{\nu}\sin\nu)\frac{n\,(1 + e\cos\nu)}{(1 - e^2)^{3/2}} \tag{4}$$

In the case of zero eccentricity, the $\dot{\nu}$ terms reduce to n, the $\ddot{\nu}$ terms vanish and the familiar Hill's equations are obtained. The solution to Hill's equations may be found through a standard Laplace transformation.[10] The relative position for circular reference orbits is given below as a function of time:

$$x(t) = \frac{\dot{x}_0}{n}\sin(nt) - \left(3x_0 + 2\frac{\dot{y}_0}{n}\right)\cos(nt) + \left(4x_0 + 2\frac{\dot{y}_0}{n}\right) \tag{5}$$

$$y(t) = \left(6x_0 + 4\frac{4\dot{y}_0}{n}\right)\sin(nt) + \frac{2\dot{x}_0}{n}\cos(nt) - (6nx_0 + 3\dot{y}_0)\,t + \left(y_0 - \frac{2\dot{x}_0}{n}\right) \tag{6}$$

$$z(t) = z_0\cos(nt) + \frac{\dot{z}_0}{n}\sin(nt) \tag{7}$$

where the initial state is defined as $[x_0, y_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0]^\top$ and t is measured from the time at which the initial state is defined. The velocity terms are found by simply taking the time-derivative of each position equation.
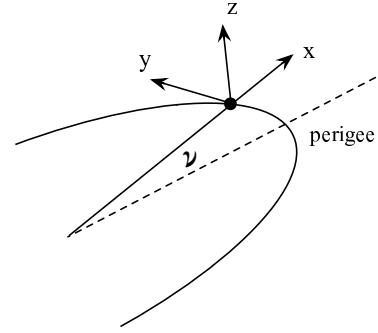
In the general case where $e > 0$, the solution is presented as a function of true anomaly. We use the procedure outlined in the journal article by Inalhan[8] to solve for the relative motion in eccentric orbits. The relative state is expressed at any true anomaly as a linear combination of 6 integration constants.

$$
\begin{bmatrix}
x(\nu) \\
y(\nu) \\
x'(\nu) \\
y'(\nu) \\
z(\nu) \\
z'(\nu)
\end{bmatrix}
=
\begin{bmatrix}
r_1(\nu) \\
r_2(\nu) \\
r_3(\nu) \\
r_4(\nu) \\
r_5(\nu) \\
r_6(\nu)
\end{bmatrix}
\begin{bmatrix}
d_1 \\
d_2 \\
d_3 \\
d_4 \\
d_5 \\
d_6
\end{bmatrix}
\equiv R(\nu)D
\tag{8}
$$

Given an initial relative state $X(\nu_0)$, the state at any other point in the orbit, $X(\nu)$, is determined by first computing the integration constant vector D, then computing the R matrix at the desired true anomaly.

$$
\begin{aligned}
D &= R(\nu_0)^{-1}X(\nu_0) \\
X(\nu) &= R(\nu)D
\end{aligned}
\tag{9}
$$

5. Geometric Parameters for Periodic Relative Trajectories

Accurate solutions to the relative motion enable us to identify families of relative trajectories suitable for several different types of formation flying missions. In general, a formation is realized through the combination of individual trajectories. From a guidance standpoint, we seek to define naturally repeating trajectories so that (in the absence of noise, disturbances and modeling errors) the relative motion is maintained with zero control effort. In other words, the relative motion is "T-periodic", repeating once each orbit period. From an orbital element standpoint, the repeating trajectories must have zero semi-major axis difference.

We wish to define the target trajectory of each satellite in terms of geometric properties, to facilitate a direct mapping to the high-level formation objectives. Two different methods have been developed for parameterizing relative trajectories as a function of geometric components. The first method applies to circular orbits, the second to eccentric orbits. In both cases, the in-plane (x-y) and out-of-plane (z) motion are decoupled.

5.1. Relative Motion Geometry for Circular Orbits

In the case of circular orbits, the geometry of the relative motion is easily expressed as a superposition of along-track offset, in-plane elliptical motion, and cross-track oscillation. The following five parameters are used to fully define the geometry of any type of relative trajectory: An example trajectory is shown in

Table 1. Geometric Parameters for Relative Motion in Circular Orbits

Parameter	Description
y_0	Along-track offset. Defines the center of the in-plane relative ellipse.
a_E	Semi-major axis of relative ellipse.
β_0	Phase angle on relative ellipse at ascending equator crossing. Measured positively from $-x$ axis to $+y$ axis of Hill's frame.
z_i	Cross-track amplitude due to inclination difference.
z_Ω	Cross-track amplitude due to right ascension difference.

Figure 4 on the following page, illustrating the separate in-plane and out-of-plane motions. The relative ellipse is a 2×1 elliptical shape in the x-y plane that is achieved by a small eccentricity difference. The phase angle β_0 defines the location of the satellite on this ellipse at the moment when the absolute orbit crosses the equator. The center of this ellipse is located at the along-track offset, y_0. When the semi-major axis a_E is zero, the elliptical motion vanishes, and the satellite remains fixed at y_0. This is often referred to
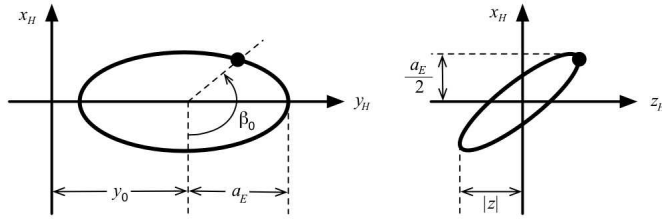
Figure 4. Example of Relative Motion in Circular Orbits

as a leader-follower configuration. The amplitude of the cross-track oscillation is defined as:

$$|z| = \sqrt{z_i{}^2 + z_\Omega{}^2} \tag{10}$$

These parameters are chosen for the cross-track motion because the presence of inclination difference can result in significant secular drift from the J_2 perturbation. Thus, it is often desirable to achieve the desired cross-track amplitude with as little inclination difference as possible, so that $z_\Omega > z_i$.

Hill's frame coordinates may be computed from these geometric parameters through the following non-linear transformation:

$$x = -\frac{1}{2} a_E \cos(\alpha_0 + \theta) \tag{11}$$

$$y = y_0 + a_E \sin(\alpha_0 + \theta) \tag{12}$$

$$z = z_i \sin(\theta) - z_\Omega \cos(\theta) \tag{13}$$

$$\dot{x} = \frac{1}{2} a_E n \sin(\alpha_0 + \theta) \tag{14}$$

$$\dot{y} = a_E n \cos(\alpha_0 + \theta) \tag{15}$$

$$\dot{z} = n \left(z_i \cos(\theta) + z_\Omega \sin(\theta) \right) \tag{16}$$

where θ is the true latitude, and the angle α_0 is related to the original phase angle β_0 through the equation:[11]

$$\alpha_0 = \sin^{-1} \left(\frac{\sin \beta_0}{\sqrt{4 - 3 \sin^2 \beta_0}} \right) \tag{17}$$

The inverse transformation, converting from Hill's coordinates to geometric parameters, is found through simple algebraic manipluation.[12]

Referring back to Eq. 6 on page 6, we can identify the constraint for periodic motion. It is equivalent to setting the only coefficient on t equal to zero, to ensure that the along-track separation does not grow unbounded with time.

$$6 n x_0 + 3 \dot{y}_0 = 0 \tag{18}$$

Combining Equations 11 and 15 into Eq. 18, we find that the constraint is satisfied.

5.2. Relative Motion Geometry for Eccentric Orbits

The relative motion occurring in eccentric orbits is fundamentally different from that found in circular orbits. The in-plane motion no longer follows a 2×1 ellipse, and the cross-track oscillation is not necessarily centered about the origin. The shape of the trajectory depends upon the eccentricity, as well as the points in the orbit where the maximum radial and cross-track amplitudes occur.

As discussed in Section 4 on page 6, the method of predicting the relative motion in eccentric orbits first requires a set of integration constants to be found. The most straightforward way to compute the integration constants, D, is to fully constrain the position and velocity at a particular true anomaly (as in Eq. 9 on the preceding page). However, one may elect to introduce other types of constraints. The geometric parameters in Table 2 on the next page provide enough information to define a new set of constraints that can be used

Table 2. Geometric Parameters for Relative Motion in Eccentric Orbits

Parameter	Description
y_0	Center of along-track motion
$\bar{x}$	Maximum radial amplitude
$\nu_{\bar{x}}$	True anomaly where $\bar{x}$ occurs
$\bar{z}$	Maximum cross-track amplitude
$\nu_{\bar{z}}$	True anomaly where $\bar{z}$ occurs

to compute the integration constants for a desired trajectory. Examples of two different relative trajectories are shown in Figure 5. For each trajectory, $y_0 = \bar{x} = \bar{z} = 1$. For the trajectory on the left: $e = 0.7$, $\nu_{\bar{x}} = 90^0$, and $\nu_{\bar{z}} = 180^0$. On the right, $e = 0.7$, $\nu_{\bar{x}} = 90^0$, and $\nu_{\bar{z}} = 126.9^0$. This value was chosen for $\nu_{\bar{z}}$ because it corresponds to an eccentric anomaly of 90^0, which results in symmetric cross-track motion.
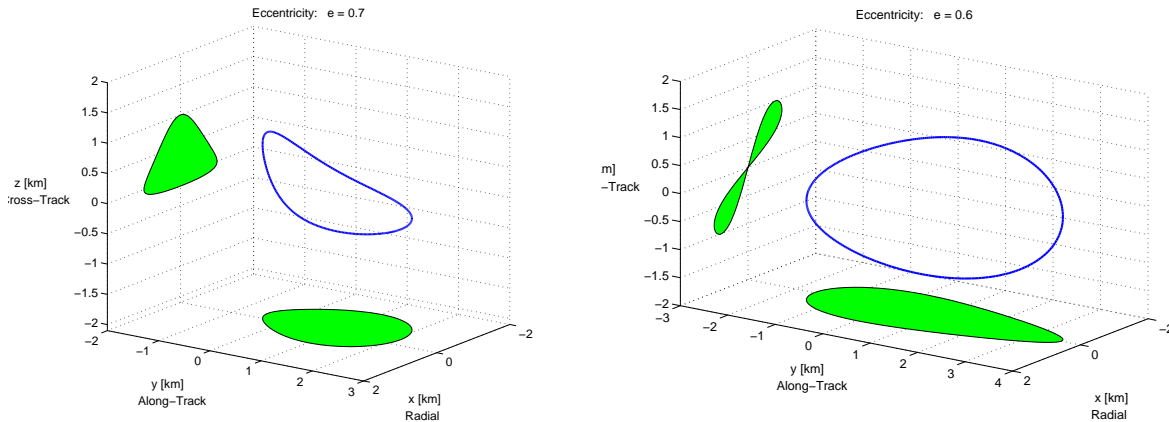
Figure 5. Example Periodic Relative Trajectories in Eccentric Orbits

A total of six constraint equations are required. The five independent parameters account for five of the equations. The remaining equation is obtained from the periodicity (zero-drift) constraint. Enforcing zero-drift in an eccentric orbit is achieved by meeting the following constraint:[8]

$$y'(\nu)/x(\nu) = -\frac{1+e}{2+e} \tag{19}$$

Let us examine the constraints imposed by each parameter. First, $\bar{x}$ is defined as the maximum radial amplitude, occurring at a true anomaly of ν_x. Together these bring two constraint equations – at ν_x, the x position must equal $\bar{x}$ and the x velocity must equal 0. A similar set of constraints are obtained from $\bar{z}$ and ν_z. The y_0 parameter is defined as the center of motion in the y (along-track) direction. Thus, it can be expressed as follows:

$$y_0 = \frac{1}{2}\Big(y(\nu_{y-}) + y(\nu_{y+})\Big) \tag{20}$$

where ν_{y-} and ν_{y+} are the true anomaly values corresponding to the minimum and maximum values of y, respectively. These angles are not known a priori, so they must be determined by some method. The resulting set of constraints may be expressed as:

$$\begin{bmatrix} \bar{x} \\ y_0 \\ 0 \\ 0 \\ \bar{z} \\ 0 \end{bmatrix} = \begin{bmatrix} r_1(\nu_{\bar{x}}) \\ \frac{1}{2}\left(r_2(\nu_{y-}) + r_2(\nu_{y+})\right) \\ r_3(\nu_{\bar{x}}) \\ -(2+e)/(1+e)r_1(0) - r_4(0) \\ r_5(\nu_{\bar{z}}) \\ r_6(\nu_{\bar{z}}) \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{bmatrix} \tag{21}$$

This approach is considered the "bounding" method for trajectory definition, because the x and z motions are bounded by the chosen parameters.

A double-iterative procedure is used to compute ν_{y-} and ν_{y+}. First, initial guesses for ν_{y-} and ν_{y+} are made. The initial guesses are chosen to be $\nu_{\bar{x}} \pm \pi/2$. These values are then used to define the constraint equation for y_0. All other constraint equations are already fully defined, which allows the D vector to be computed. This represents the first level of iteration. Since D defines the entire trajectory, the actual minimum and maximum y values may be computed, along with the true anomalies at which they occur. This computation, which is the second level of iteration, is performed using the Newton-Raphson method with the first and second derivatives of y with respect to ν. If the extreme true anomalies are not close enough to the previous guess, then ν_{y-} and ν_{y+} are updated and the iteration continues. The approach has shown to work accurately and extremely fast in practice, always returning a valid solution in only 1-5 iterations and taking only a few tenths of a second.

If desired, a slightly different set of geometric parameters may be used. Rather than choosing the size and location of the maximum radial amplitude, one may instead choose a particular $[x, y]$ location in the relative frame, and the true anomaly where that position is to occur. Similarly, one may elect to define a particular z value, along with the true anomaly at which it occurs, instead of defining the value of the cross-track amplitude. The constraint equations associated with this alternate "positioning" method are given below:

$$
\begin{bmatrix} x \\ y_0 \\ y \\ 0 \\ z \\ 0 \end{bmatrix} = \begin{bmatrix} r_1(\nu_{xy}) \\ \frac{1}{2}\left(r_2(\nu_{y-}) + r_2(\nu_{y+})\right) \\ r_2(\nu_{xy}) \\ -(2+e)/(1+e)r_1(0) - r_4(0) \\ r_5(\nu_z) \\ r_6(\nu_{\bar{z}}) \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{bmatrix} \tag{22}
$$

This positioning method is used to define a repeating tetrahedron formation in Section 5.2.1.

Each of the two goal sets provide a full set of constraint equations, so that only one solution exists. Another approach would be to partially constrain the problem, define a cost function to be minimized, and then use a linear program to find an optimal solution. This type of technique was recently presented where the cost function is defined as the 1-norm of the weighted error in the element differences.[13] The weights can be chosen to approximate the relative contribution from each element error to the required delta-v. The inherent drawback with an element-based initialization method is that it does not explicitly bound the geometric properties of the trajectory. Therefore, the approach is most useful in a formation maintenance sense, where the individual trajectories are already "close" to the desired formation geometry.

5.2.1.   Defining a Tetrahedron Geometry

NASA has identified a benchmark problem for highly elliptic orbits in which four spacecraft repeatedly achieve a regular tetrahedron formation at apogee.[14] The reference follows a 1.2 by 18 Earth radii orbit, corresponding to an eccentricity of 0.875 and a semi-major axis of about 61.2 km. At apogee, the four spacecraft occupy the four points of the tetrahedron, with a common separation of 10 km between all spacecraft. The orientation of the tetrahedron is arbitrary. An additional objective is that the spacecraft avoid close approaches (less than 1 km) throughout the orbit.

The initialization of the target trajectories for this formation can be achieved using the "positioning" method outlined in Eq. 22. One spacecraft serves as the reference, leaving three trajectories to be defined. For each of the three relative spacecraft, we specify the desired position in the x, y, and z axes at a true anomaly of $\nu = 180$ deg. Table 3 on the next page summarizes one possible set of geometric parameters for each trajectory. In this case, we specify only one trajectory to have an out-of-plane component. The relative trajectories associated with these geometric parameters are shown in Figure 6 on the following page. The plots verify that the expected tetrahedron geometry occurs at apogee. The closest distance between any two spacecraft over the course of the orbit is 3.7 km. These trajectories represent just one orientation – an infinite number of orientations are possible. One may rotate the $[x, y, z]^{\top}$ vector of each non-reference point in the tetrahedron through a common set of Euler angles to achieve a different orientation. This procedure may be used in an offline analysis in which the orientation of the tetrahedron and the along-track center y_0 are varied, and the resulting minimum and average separation distances are measured. The analysis could

Table 3. Geometric Parameters for the Eccentric Tetrahedron Formation

Trajectory	y_0 [km]	x [km]	y [km]	ν_{xy} [deg]	$\bar{z}$ [km]	$\nu_{\bar{z}}$ [deg]
A	0	0	10	180	0	180
B	0	$5\sqrt{3}$	5	180	0	180
C	0	$5/\sqrt{3}$	5	180	$10\sqrt{2/3}$	180
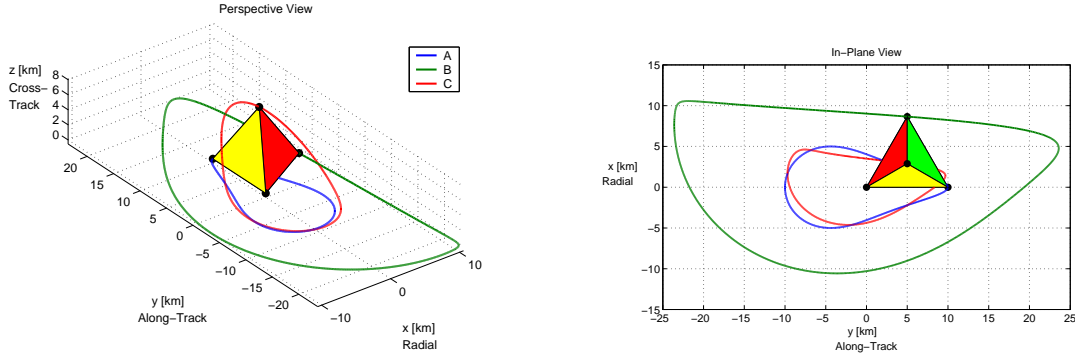
Figure 6. Trajectories for the Eccentric Tetrahedron Geometry

be taken a step further by considering the known gravitational perturbations and disturbances, so that the formation may be designed to minimize the impact of these effects.

6.   Distributed Guidance Law

Formations are designed by combining several relative trajectories in such a way that a desired geometric configuration is achieved. In order to realize a formation, each spacecraft must achieve one of the relative trajectories. It is typically not important which spacecraft achieves which trajectory, as long as all desired trajectories are met so that the overall formation geometry is realized. Therefore, the main function of the formation guidance law is that of trajectory assignment. The objective is to achieve all desired trajectories in such a way that minimizes the total cost.

The guidance law designed for the DFF system operates at both the team and cluster level. The set of target states for an individual team or for the entire cluster may be defined by the operator according to three different levels of autonomy:

- **Low Autonomy** – The target state for each spacecraft is defined explicitly. It may be defined in Hill's frame coordinates, orbital element differences, or in terms of the geometric parameters. The control law immediately tracks the new desired trajectory.

- **Medium Autonomy** – A set of geometric goals is supplied, where each goal set corresponds to a desired trajectory. The guidance law works to find the optimal configuration, assigning each target to a different member such that the total cost is minimized.

- **High Autonomy** – A few high-level objectives are supplied, such as a pre-defined formation type, size, and orientation. The geometric goals for the team/cluster are computed online to meet the given objectives. Each target is then assigned to a different member such that the total cost is minimized.

The geometric goals are described in Tables 1 and 2 for circular and eccentric orbits.

The guidance law is not used in the "Low Autonomy" case, as the target states are assigned directly from the operator. At the medium and high autonomy levels, however, the targets are assigned on-orbit. The general procedure is as follows:

1. The geometric goals for the team are either supplied to or computed by the team captain.

2. The captain distributes the team goals to all relative team members.

3. Each recipient uses the control law to estimate the cost to achieve all desired trajectories.

4. The vector of costs from each relative member is returned to the captain.

5. The captain assembles all cost vectors into a single cost matrix.

6. The captain applies an assignment algorithm to the cost matrix to find a solution that minimizes the total cost.

7. The captain sends out the newly assigned geometric goals to each relative team member.

The guidance law is therefore both distributed and centralized. The cost estimation is distributed across all satellites in the team, but the assignment task is performed centrally, by the captain.

The total cost is defined as a weighted sum of the total delta-v's for each spacecraft to achieve its target state. The individual delta-v's are weighted according to their remaining fuel percentage, in order to promote equal fuel usage throughout the cluster. The cost for the ith satellite to reach the jth target state is:

$$c_{i,j} = f_i^{-x} \times \Delta V_{ij} \qquad (23)$$

where f_i is the remaining fuel percentage of the ith spacecraft, and $x > 0$ is an adjustable parameter indicating the importance of fuel equalization. The general problem is to assign N target states to M satellites such that the total cost is minimized. This results in a $M \times N$ cost matrix. In the case where $N \neq M$, the cost matrix is made square by adding rows or columns whose elements are much larger than the maximum value of the original matrix.

Two different approaches to solving the assignment problem have been implemented within the Guidance Law. The first approach, termed the "optimal method", involves searching over all possible permutations to find the one with the minimum total cost. The total number of unique permutations is $N!$ for a square cost matrix of size N. This approach is therefore computationally cumbersome as N becomes large, i.e., ≥ 8. The advantage is that a globally optimal solution is guaranteed.

The second approach is called the "privileged method". This technique requires considerably less computation, but does not guarantee that a globally optimum solution is found. It consists of the following steps:

1. Determine the minimum projected cost of each satellite.

2. Determine which satellite has the highest minimum cost.

3. Assign that satellite to the target state corresponding to its minimum cost.

4. Repeat steps 1-2 for all remaining members and remaining target states.

A similar technique is presented by Tillerson.[15] The optimal and privileged approaches are compared in Section 6.2 on the next page below.

Whichever method is used, the ground operator has the flexibility of restricting the target state distribution. Each target state may be restricted so that it is considered by only a specific subset of team members. This is accomplished in the cost estimation stage by forcing $c_{i,j}$ to be an extremely large number for cases where the jth target state is not allowed to be assigned to the i_{th} satellite. The assignment algorithm naturally avoids these high cost combinations.

The guidance law has also been designed to take advantage of the additional freedom available in circular reference orbits. Here, we allow two types of target states to be identified: *fixed* and *variable*. With the fixed target state, we specify both the desired state and the point in the orbit at which it is to occur. With the variable target state, we consider all possible states along the trajectory. When multiple target states are defined along the same trajectory, the original phase separation is maintained as a constraint. The details of this "variable state" method require a lengthy discussion and are beyond the scope of this paper.

In eccentric orbits, only fixed target states are possible. This is due to the restricted nature of the relative trajectories in eccentric orbits, where each state along a given trajectory must occur at a particular true anomaly. In the cost estimation stage of the guidance law, the fixed target state is defined by using the true anomaly associated with the longest allowable maneuver duration.

6.1. Distributed Assignment in a Multiple-Team Setting

When assigning target states to a cluster composed of multiple teams, they are initially uploaded to the cluster reference. They are then transmitted to all relatives in the top-level team. If any of these relatives also serves as a reference for a lower-level team, it transmits the targets to its relatives, and so on in a recursive manner until the targets have been sent to all spacecraft in the cluster. The target states are all defined with respect to the cluster reference, so the relative state of each spacecraft must also be computed in the same frame. Therefore, along with the target state information, each reference also sends its relative state with respect to the cluster reference. This enables each satellite to compute its relative state in the cluster frame. Consider the ith spacecraft in the cluster, whose reference is spacecraft j. Let the C subscript denote the cluster frame, and T denote the team frame. The relative state of the ith spacecraft in the cluster frame is then:

$$x_{C,i} = x_{C,j} + x_{T,i} \qquad (24)$$

where $x_{C,j}$ is transmitted from spacecraft j, and $x_{T,i}$ is readily available from the Coordinate Transformation module.

Once each satellite has the target states and its relative state in the cluster frame, it computes the set of costs to achieve all targets, and transmits the cost vector to back to its reference. Each reference in the hierarchy compiles the cost vectors from its relatives and then sends the data up to its reference, until finally the cluster reference has the complete set of cost data from all satellites in the cluster. The privileged assignment method is then used to quickly determine the best possible assignment, and the targets are distributed accordingly throughout the teams. The final step is for each relative satellite to transform the target state to its team-based coordinate frame before reconfiguring to the new trajectory.

6.2. Example Assignment Problem

Consider a team with 8 relative spacecraft flying in a circular orbit at 550 km altitude. The team is initialized in a leader-follower formation with a common along-track spacing of 200 m. Each spacecraft also has an out-of-plane component so that they follow the same ground-track. We wish to reconfigure the formation so that two equally phased projected-circle geometries are achieved, one with a 1 km radius and the other with a 500 m radius. In both cases, the reference is located at the center of the circle. The first step is to estimate the cost for each satellite to achieve all possible trajectories. In this case, all desired states lie on two projected circles. We therefore define the targets as two sets of four equally phased "variable states".

Each satellite uses the analytic control law to estimate the total delta-v required to reach the new trajectory. Using an angular resolution of 5 deg, the computation is performed for a total of 144 possible target states (72 for each trajectory). The final cost for each satellite is then weighted according to its remaining fuel percentage. In this example, the fuel weighting exponent is $x = 1$, and the ith satellite has a remaining fuel percentage of:

$$f_i = 10 \times (1 + i)\% \qquad (25)$$

The cost estimates for each satellite are then submitted to the captain, who performs the assignment. The results from both the privileged and optimal assignment method are compared in Table 4. The privileged

Table 4. Optimal vs. Privileged Assignment Methods: Cost and Computation Time

Method	Weighted Cost	Delta-V [m/s]	Computation Time [sec][*]
Privileged	16.28	8.30	0.004
Optimal	16.22	8.16	301.8

[*] This example was run in MATLAB version 6.5.1 on a Mac PowerBook G4 running OS X.

assignment method finds a solution that is just 0.6% higher than the optimal result, and it provides a huge savings in computation time. The optimal method takes a considerable amount of time to run because it is comparing the cost associated with $8! = 40,320$ unique combinations. The worst possible configuration had a weighted cost of 21.3 m/s, which is about 30% higher than the optimal solution.

It is interesting to note the effect of weighting the costs. The four spacecraft with the lowest remaining fuel percentages (20, 30, 40 and 50%) are assigned to the smaller circle, while those with more fuel are assigned to the larger circle. This is due to the fact that the larger circle requires twice the out-of-plane component as the smaller, and the delta-v is extremely sensitive to out-of-plane changes. The relative motion associated with this reconfiguration maneuver is shown in Figure 7. The initial and final locations are marked. The ellipses in the x-y plane show that the in-plane motion is achieved first. A single cross-track burn applied at either end of the ellipse then puts each spacecraft on the projected-circle.
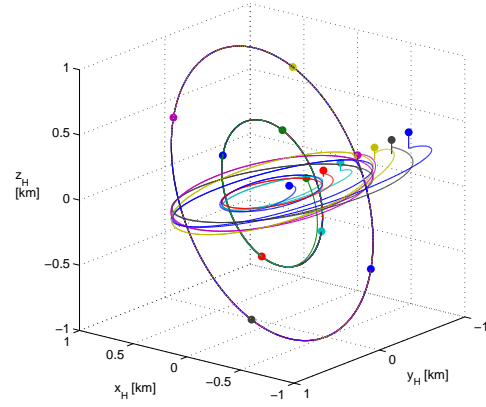

Figure 7. Trajectories of the Reconfiguration

6.3. Transitioning Between Circular and Eccentric Goal Sets

The Guidance Law and Control Law modules are designed to accommodate both circular and eccentric orbits. An eccentricity tolerance is defined at startup, and may be adjusted at any time by the ground station. If the reference orbit eccentricity is below this tolerance, the orbit is considered circular. If it is at or above the tolerance, it is considered eccentric.

For nearly circular orbits, the eccentricity may be close to the specified tolerance. The ground station may continue to command circular geometric goals even though the eccentricity has exceeded the tolerance, or vice versa. In such cases, the software must be able to dynamically switch the geometric goals between the circular and eccentric sets. The parameter y_0 has the same meaning for both sets. The maximum radial amplitude $\bar{x}$ is simply equal to half of the semi-major axis of the relative ellipse:

$$\bar{x} = a_E/2 \tag{26}$$

The maximum cross-track amplitude is simply the magnitude of the two independent variables, z_i and z_Ω:

$$\bar{z} = \sqrt{z_i^2 + z_\Omega^2} \tag{27}$$

In order to define the true anomaly location for $\bar{x}$, first recall that the phase angle β_0 is defined as the rotation from the negative x-axis to the positive y-axis. Therefore, the maximum *positive* radial amplitude is defined at $\beta = 180$ deg. By definition, this phase angle occurs at zero true latitude ($\theta = 0$). Noting that $\theta = \omega + \nu$, and transforming β_0 to α_0 with Eq. 17 on page 8, we find that $\bar{x}$ occurs at the true anomaly:

$$\nu_{\bar{x}} = \pi - \alpha_0 - \omega \tag{28}$$

The true latitude where $\bar{z}$ occurs is found by setting $\dot{z} = 0$ (see Eq. 16 on page 8). The true anomaly is then found by subtracting the argument of perigee.

$$\nu_{\bar{x}} = \tan^{-1}\left(z_i/z_\Omega\right) - \omega \tag{29}$$

7. Control Law Design

Formation flying maneuvers can be grouped into two different categories: *reconfiguration* and *maintenance*. In a reconfiguration, the desired trajectory of the spacecraft is changed, so that the spacecraft maneuvers from its original trajectory to the new one. These are typically "large-scale" maneuvers, and can potentially require a significant amount of delta-v. Maintenance maneuvers, on the other hand, typically involve small corrections to eliminate small errors in the relative state. Here, the spacecraft returns to its original trajectory after disturbances have caused it to drift away.

The control objective of each type of maneuver is the same. We wish to proceed from an initial state to a target state while expending the least amount of control effort possible. In addition, we may seek to

minimize the maneuver duration. A maneuver planning method has been developed for the DFF system which uses a model predictive controller to provide a time-weighted, fuel-optimal control solution to this problem. The application of model-predictive control techniques to the problem of relative orbit control has been presented in various forms in recent years.[15–18] In general, a linear program is used to compute an impulsive delta-v sequence over a fixed time window, such that the initial state is brought to the desired state at the final time.

It is worth noting that this control approach involves the intermittent application of maneuvers, with no pre-defined control frequency. It is well-suited for reconfiguration maneuvers and for coarse formation keeping. In general, it is sensible to apply this approach when the amount of time between maneuvers is much greater than the average maneuver duration. The ability to meet this condition depends upon the magnitude of differential disturbances and the control requirements. It is not intended, therefore, for precision formation control, where continuous corrections are required to maintain tight position error tolerances. Under these circumstances, an active control method, such as LQG for example, should be used.[19]

7.1. Model Predictive Control Formulation

In a previous paper,[12] two types of model predictive controllers were presented for planning maneuvers in circular and eccentric orbits. The relative dynamics for each method are expressed in the cartesian Hill's frame. This approach is found to work well for circular and low-eccentricity orbits, but it is not well-suited for eccentric orbits in general. In order to set up the problem for a linear program, the LTV state-space representation of the relative dynamics is discretized with a zero-order hold at each step over the entire time-span of the maneuver. The discrete matrices provide an approximation whose accuracy depends upon the size of the time-step, Δt. As $\Delta t \to 0$, the error associated with the discretization vanishes but the number of samples required for the linear program goes to infinity. The sensitivity of this error to the time-step increases with eccentricity. For a practical application of the linear program, considering memory limitations and processor speed, a maximum number of samples must be respected. Therefore, a certain amount of discretization error is inescapable. For highly eccentric orbits, this error becomes unacceptable.

It is preferable to express the relative dynamics using Gauss' variational equations (GVEs), where the relative state is defined as a set of differential orbital elements. There exist inherent advantages in using GVEs for relative orbit dynamics and control.[13] Linearizing about the orbital elements allows a large range of relative motion to be accurately captured with small perturbations in the relative state. This curvilinear coordinate frame therefore imposes smaller linearization errors than the rectilinear frame used by Hill's and Lawden's equations. In addition, the discrete state-transition matrix for differential orbital elements is accurate for any size time-step, which facilitates a more practical application of linear programming.

Gauss' variational equations relate accelerations applied in the along-track, radial, and cross-track direction to the instantaneous time-derivative of the orbital elements.[10] Let us define the standard element set[a] as $\boldsymbol{e_s} = [a, i, \Omega, \omega, e, M]^{\top}$, where a is the semi-major axis, i is the inclination, Ω is the longitude of the ascending node, ω is the argument of perigee, e is the eccentricity and M is the mean anomaly. The state-space form of the equations is provided below:

$$\frac{d}{dt}\begin{bmatrix} a \\ i \\ \Omega \\ \omega \\ e \\ M \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ n \end{bmatrix} + \frac{1}{h}\begin{bmatrix} 2a^2 e \sin\nu & 2a^2 p/r & 0 \\ 0 & 0 & r\cos\theta \\ 0 & 0 & r\sin\theta/\sin i \\ -p\cos\nu/e & (p+r)\sin\nu/e & -r\sin\theta\cot i \\ p\sin\nu & (p+r)\cos\nu + re & 0 \\ \eta p\cos\nu/e - 2r & -\eta(p+r)\sin\nu/e & 0 \end{bmatrix}\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \tag{30}$$

where n is the mean orbit rate, p is the semi-latus rectum, r is the magnitude of the radius vector, h is the angular momentum, and $\eta = \sqrt{1 - e^2}$. The equation can be compactly written as:

$$\dot{\boldsymbol{e_s}} = \bar{A}(\boldsymbol{e_s}) + B(\boldsymbol{e_s})\boldsymbol{u} \tag{31}$$

We are interested in the dynamics of the relative state, which is the differential element set defined as:

$$\boldsymbol{\delta e_{si}} = \boldsymbol{e_{si}} - \boldsymbol{e_s} \tag{32}$$

---

[a]The "s" subscript differentiates the standard element set from a different set defined in Section 7.2 on page 17.

where $\boldsymbol{e_{si}}$ is the element set of the ith spacecraft, and $\boldsymbol{e_s}$ is the element set of its reference. Differentiating Eq. 32 on the previous page with a first order approximation yields the linearized relative dynamics:

$$\dot{\boldsymbol{\delta e}}_{si} = A(\boldsymbol{e_s}) + B(\boldsymbol{e_s})\boldsymbol{u} \tag{33}$$

where the matrix $A(\boldsymbol{e_s})$ is found by taking the partial derivative of the vector $\bar{A}(\boldsymbol{e_s})$ with respect to $\boldsymbol{e_s}$. This vector has only one non-zero element, $n = \sqrt{\mu/a^3}$, which is clearly only a function of the semi-major axis a. This causes $A(\boldsymbol{e_s})$ to be a 6×6 matrix of zeros, except for the bottom left element, which is $\partial \dot{M}/\partial a = -3n/2a$.

The continuous-time system is discretized using a zero-order hold over a variable time-step Δt, which is defined by an even spacing of N true anomaly values. Because the system is LTV, the zero-order hold operation must be carried out at every k^{th} time-step to obtain:

$$\boldsymbol{x}_{k+1} = A_k \boldsymbol{x}_k + B_k \boldsymbol{u}_k \tag{34}$$

For each $k \in (1, N)$, a unique set of discrete A_k and B_k matrices are obtained. Although the continuous-time A matrix is not a function of true anomaly, the discrete version of the matrix changes over time because of the variation in Δt. The equations for the first two states are expressed as follows:

$$\begin{aligned} \boldsymbol{x}_1 &= A_0 \boldsymbol{x}_0 + B_0 \boldsymbol{u}_0 \\ \boldsymbol{x}_2 &= A_1 \boldsymbol{x}_1 + B_1 \boldsymbol{u}_1 \\ &= A_1 A_0 \boldsymbol{x}_0 + A_1 B_0 \boldsymbol{u}_0 + B_1 \boldsymbol{u}_1 \end{aligned} \tag{35}$$

Extending to the N_{th} state, we find:

$$\begin{aligned} \boldsymbol{x}_N &= \begin{bmatrix} A_{N-1} & A_{N-2} & \cdots & A_0 \end{bmatrix} \boldsymbol{x}_0 \\ &+ \begin{bmatrix} A_{N-1} & A_{N-2} & \cdots & A_1 \end{bmatrix} B_0 \boldsymbol{u}_0 \\ &+ \begin{bmatrix} A_{N-1} & A_{N-2} & \cdots & A_2 \end{bmatrix} B_1 \boldsymbol{u}_1 \\ &+ \vdots \\ &+ B_{N-1} \boldsymbol{u}_{N-1} \end{aligned} \tag{36}$$

Let us now define the following:

$$\begin{aligned} \acute{A} &= \begin{bmatrix} A_{N-1} & A_{N-2} & \cdots & A_0 \end{bmatrix} \\ \acute{B} &= \begin{bmatrix} \begin{bmatrix} A_{N-1} & A_{N-2} & \cdots & A_1 \end{bmatrix} B_0 & \begin{bmatrix} A_{N-1} & A_{N-2} & \cdots & A_2 \end{bmatrix} B_1 & \cdots & B_{N-1} \end{bmatrix} \\ \acute{u} &= \begin{bmatrix} \boldsymbol{u}_0 & \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_{N-1} \end{bmatrix}^\top \end{aligned} \tag{37}$$

We may now compactly express the Nth state as follows:

$$\boldsymbol{x}_N = \acute{A}\boldsymbol{x}_0 + \acute{B}\acute{u} \tag{38}$$

This gives us an expression for the Nth state in terms of the initial state, $\boldsymbol{x}_0$, and the control history, $\boldsymbol{u}_k$, for $k = 0 \rightarrow N - 1$. The objective is to find a control history that requires the minimum cumulative delta-v, subject to the constraint that the desired state is achieved at $k = N$.

The terminal constraint may be written as:

$$|\boldsymbol{x}_N - \boldsymbol{x}^*| \le \boldsymbol{\epsilon} \tag{39}$$

for a sufficiently small $\boldsymbol{\epsilon}$ vector. Noting that the right-hand side is an absolute value, we may rewrite the above expression as two inequalities.

$$\begin{aligned} \boldsymbol{x}_N - \boldsymbol{x}^* &\ge -\boldsymbol{\epsilon} \\ \boldsymbol{x}_N - \boldsymbol{x}^* &\le +\boldsymbol{\epsilon} \end{aligned} \tag{40}$$

Substituting Eq. 38 into Eq. 40, we obtain:

$$A^N \boldsymbol{x}_0 + \acute{B}\acute{\boldsymbol{u}} - \boldsymbol{x}^* \geq -\boldsymbol{\epsilon}$$
$$A^N \boldsymbol{x}_0 + \acute{B}\acute{\boldsymbol{u}} - \boldsymbol{x}^* \leq +\boldsymbol{\epsilon}$$

(41)

Further algebraic manipulation yields:

$$
\begin{aligned}
-\acute{B}\acute{\boldsymbol{u}} &\leq & \boldsymbol{\epsilon} + A^N \boldsymbol{x}_0 - \boldsymbol{x}^* \\
\acute{B}\acute{\boldsymbol{u}} &\leq & \boldsymbol{\epsilon} - A^N \boldsymbol{x}_0 - \boldsymbol{x}^*
\end{aligned}
$$

(42)

Now let $\tilde{A}$ and $\tilde{\boldsymbol{b}}$ be defined as:

$$\tilde{A} = \begin{bmatrix} -\acute{B} \\ \acute{B} \end{bmatrix}$$

(43)

$$\tilde{\boldsymbol{b}} = \begin{bmatrix} \boldsymbol{\epsilon} + \acute{A}\boldsymbol{x}_0 - \boldsymbol{x}^* \\ \boldsymbol{\epsilon} - \acute{A}\boldsymbol{x}_0 + \boldsymbol{x}^* \end{bmatrix}$$

so that the inequality may be written as follows:

$$\tilde{A}\acute{\boldsymbol{u}} \leq \tilde{\boldsymbol{b}}$$

(44)

The problem is now cast as a linear program, which is solved using the Simplex method. The objective is to minimize the cost $\tilde{\boldsymbol{c}}\acute{\boldsymbol{u}}$ subject to the constraint defined in Eq. 44. Here, the cost coefficient vector, $\tilde{\boldsymbol{c}}$, has $3N$ columns, and is nominally composed of all 1's. The weights may be adjusted to provide a greater or lesser penalty at different times, or even for different axes in the relative frame.

When planning a maneuver, we search for a control solution that represents the combination of minimum-time *and* minimum-fuel objectives. First, the the control trajectory is computed several times, over a range of possible maneuver durations defined by the ground operator. The nominal cost for each case, c_i, is defined as the total delta-v required for the maneuver. This cost is then weighted by the maneuver duration, T_i, so as to penalize the longer maneuvers. An exponential weighting method is used in this system, where the weighted cost w_i is computed as:

$$w_i = c_i \times \left(\frac{T_i}{T_{min}} \right)^x$$

(45)

where T_{min} is the minimum maneuver duration considered, and $x \geq 0$ is an adjustable parameter. Setting $x = 0$ causes the duration to be ignored. In this case, only the longest possible duration would be considered, since it must correspond to the minimum delta-v. Increasing x causes greater importance to be placed on the maneuver duration.

7.2. Relative Frame Transformations

As detailed in the previous section, the model predictive control formulation uses the relative dynamics of the standard differential elements. It is therefore necessary to determine the desired relative state in this coordinate frame. The desired trajectory is defined (either by the ground station operator or the Guidance Law) as a set of static geometric parameters. It is a straightforward process to transform this parameter set into Hill's frame coordinates (see Section 5 on page 7). The next step is to transform from Hill's coordinates to standard differential elements. This is implemented as a series of two transformations. The first is a non-linear transformation from Hill's coordinates to a non-standard set of differential elements (termed Alfriend elements hereafter), and the second is a linear rotation to the standard differential set.

Let us now introduce a non-standard set of differential elements, which we will term Alfriend elements. The set is defined as $\boldsymbol{e_a} = [a, \theta, i, q_1, q_2, \Omega]^\top$, where

$$\theta = \omega + \nu \quad q_1 = e\cos(\omega) q_2 = e\sin(\omega)$$

(46)

and ν is the true anomaly, which is related to the mean anomaly through the function:.

$$M = \mathcal{F}(e, \nu)$$

(47)

The evaluation of this function, of course, requires the solution of Kepler's equation, which can be found iteratively.[10]

The differential element sets for the standard and Alfriend elements are defined, respectively, as:

$$\boldsymbol{\delta e_s} = [\delta a, \delta i, \delta \Omega, \delta \omega, \delta e, \delta M]^\top$$

$$\boldsymbol{\delta e_a} = [\delta a, \delta \theta, \delta i, \delta q_1, \delta q_2, \delta \Omega]^\top$$

This standard element differences are found by carrying out the following two transformations:

$$\boldsymbol{\delta e_a} = [M]\,\boldsymbol{\delta x_H} \tag{48}$$

$$\boldsymbol{\delta e_s} = [T]\,\boldsymbol{\delta e_a} \tag{49}$$

where $[M]$ is the transformation matrix from Hills-frame coordinates to Alfriend element differences,[2] and $[T]$ rotates from Alfriend to standard element differences. The $[T]$ matrix is found by taking the partial derivative of $\boldsymbol{e_s}$ with respect to $\boldsymbol{e_a}$. The resulting matrix is found to be:

$$[T] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\sin(\omega)/e & \cos(\omega)/e & 0 \\ 0 & 0 & 0 & \cos(\omega) & \sin(\omega) & 0 \\ 0 & \frac{\partial M}{\partial \theta} & 0 & \frac{\partial M}{\partial q_1} & \frac{\partial M}{\partial q_2} & 0 \end{bmatrix} \tag{50}$$

The partial derivatives of M may eventually be expressed as follows:

$$\frac{\partial M}{\partial \theta} = \left(\frac{\partial M}{\partial \nu}\right)\left(\frac{\partial \nu}{\partial \theta}\right) \tag{51}$$

$$\frac{\partial M}{\partial q_1} = \left(\frac{\partial M}{\partial \nu}\right)\left[\sin(\omega)/e - \cos(\omega)\left(\frac{\partial \nu}{\partial e}\right)\right] \tag{52}$$

$$\frac{\partial M}{\partial q_2} = -\left(\frac{\partial M}{\partial \nu}\right)\left[\cos(\omega)/e + \sin(\omega)\left(\frac{\partial \nu}{\partial e}\right)\right] \tag{53}$$

$$\tag{54}$$

where $\frac{\partial \nu}{\partial \theta} = 1$. The partials of M with respect to e and ν, however, are not available analytically, and therefore must be approximated numerically. Choosing sufficiently small values for Δe and $\Delta \nu$, the following approximations may be used:

$$\frac{\partial M}{\partial e} \approx \frac{\mathcal{F}(e + \Delta e, \nu) - \mathcal{F}(e, \nu)}{\Delta e} \tag{55}$$

$$\frac{\partial M}{\partial \nu} \approx \frac{\mathcal{F}(e, \nu + \Delta \nu) - \mathcal{F}(e, \nu)}{\Delta \nu} \tag{56}$$

The inverse, $[T]^{-1}$, may be found by direct inversion of the above matrix, or analytically through a similar procedure of finding the partial derivatives of $\boldsymbol{e_a}$ with respect to $\boldsymbol{e_s}$.

7.3. Formation Maintenance in a Multiple-Team Setting

In a multiple-team setting, each of the lower-level teams has a reference who is also a relative on a higher-level team. As a relative, the spacecraft will need to occasionally perform maintenance maneuvers to stay in formation with its reference. When it performs a maneuver, this of course affects the state of its relative team members. To the relatives, this simply appears to be a growing state error. The relative should not attempt to correct this error until the reference has completed its maneuver. Therefore, when a reference spacecraft plans a maneuver, it informs its relatives that it is "active". The relatives temporarily disable their control law so that a maneuver is not planned during this time. Once the reference has completed its maneuver, it informs the relatives that it is "idle", at which point they reactivate their control law.

Another method of dealing with this issue is for the reference to share its own state error with the relatives prior to implementing its maneuver. Each relative can then add this state error to its own error, and plan an appropriate maneuver that will correct the total error. This method is currently being tested.

8.   Collision Avoidance

The manner in which collision monitoring and avoidance is conducted is an important design consideration. When the cluster is operating nominally, all spacecraft are controlling to a relative trajectory that, by design, maintains a safe distance from all other trajectories. Therefore, as long as each spacecraft continues to maintain its desired trajectory, no collisions will occur. It is not necessary, then, to monitor for collisions under nominal circumstances. There are two circumstances, however, which will result in a deviation from the nominal trajectories: 1) during a reconfiguration maneuver, and 2) in the event of a failure.

The type of collision avoidance performed in response to a failure is considered "reactionary"; this should be treated as part of the fault management plan, which is unique to every mission and therefore beyond the scope of the DFF system. The system is responsible, however, for ensuring that the forced trajectories planned during maneuvers are safe from collision. Because the control system is decentralized, with each spacecraft planning its maneuver independently, it is not possible to plan collision-free trajectories from the outset. To do so would require a centralized planning scheme. The chosen approach is to first plan the reconfiguration, then perform "preemptive" collision avoidance before carrying out the maneuver. If a possible collision is detected, the reconfiguration is aborted and the ground operator is notified.

When a reconfiguration maneuver is planned, each spacecraft broadcasts its delta-v sequence to all other spacecraft in the cluster. This data is then used onboard each spacecraft to predict the probability of collision between itself and others over the given time window. The monitoring algorithm is based on propagating uncertainty ellipsoids which correspond to levels of probability.[20] For an ellipsoid S, for example, the 3 σ relative navigation error ($\mathcal{P} = 0.997$) is propagated using the same discrete matrices as for propagating the nominal trajectory, which correspond to the ellipsoid center. Inputs $\boldsymbol{u}$ for maneuvers are easily included. Q is the uncertainty in the dynamics. A and B may be calculated for circular or eccentric orbits.

$$x_{k+1} = Ax_k + Bu$$

$$S_{k+1} = AS_kA^\top + BQB^\top$$

If the relative ellipsoid does not intersect the hard-body spacecraft volume at the origin, the probability of a collision is conservatively bounded by $1 - \mathcal{P}$. If they do intersect, successive ellipsoids at multiple $\mathcal{P}$'s are used to estimate the integral of the probability density function.

The memory required for the collision survey of the reconfiguration is dominated by storing the discrete propagation matrices. They are calculated up front for the maximum time period of any spacecraft burn and are then used for all spacecraft in the sequence. In the example from Section 6.2 on page 13, for 1133 time steps (spanning 3.6 hours) and 7 spacecraft, the memory required to store the discrete propagation matrices, the acceleration vectors for all spacecraft, and storage for the probability and minimum distance is `862,512 bytes`. The average time of calculation for one spacecraft to check the other seven trajectories (on a 1.25 GHz Powerbook G4) is 8.8 seconds. The size of the maneuver data structures which must be communicated is `19,600 bytes`. The additional information required is the relative measurements of each spacecraft to be checked, totaling `336 bytes`.

In this reconfiguration, the probability of collision is negligible. Figure 8 shows the minimum distance between spacecraft 1 and the uncertainty ellipsoids, and the relative ellipsoid evolution of spacecraft 2. The ellipsoids grow fastest in the along-track direction, as expected.
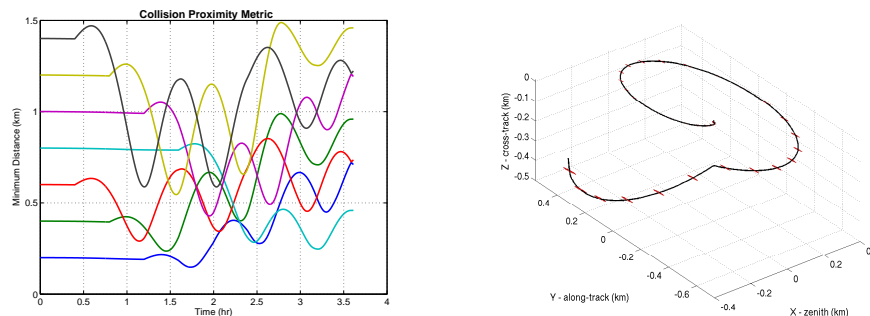

Figure 8.   Collision Monitoring Results

9. Conclusions

This paper has presented a new concept for decentralized guidance and control in formation flying missions. The hierarchical, multiple-team framework represents a compromise between wholly centralized or decentralized designs. At the team level, a distributed guidance law is implemented in which cost estimates for potential target states are computed in a decentralized manner, and the assignment of those targets is undertaken centrally by a team captain. The cost is weighted separately for each spacecraft by its remaining fuel percentage, promoting fuel equalization. The commanded target states lie on periodic relative trajectories, defined in a geometric sense to give the operator a flexible, intuitive system for designing formations. The trajectories assigned to various team members are controlled in a decentralized fashion, with each member planning its own optimal, impulsive burn sequence. Further, the cost functional of the control law is a time-weighted delta-v, enabling the combined objectives of fuel- and time-minimization.

The various methods presented here are being integrated into a decentralized formation flying (DFF) control system, implemented within a modular software architecture. The system is designed to provide several autonomous capabilities, promote robustness through software adaptability and visibility, and be extensible to large clusters. The first version of the DFF system is scheduled for release at the end of 2005.

Acknowledgments

References

[1] "MANTA product description," http://www.psatellite.com/manta, Princeton Satellite Systems.

[2] Gim, D. and Alfriend, K., "The State Transition Matrix of Relative Motion for the Perturbed Non-Circular Reference Orbit," *Proceedings of the 2001 AAS/AIAA Space Flight Mechanics Meeting*, No. AAS 01-222, Santa Barbara, CA, Feb 2001.

[3] Speyer, J. L., "Computation and Transmission Requirements for a Decentralized Linear-Quadratic-Gaussian Control Problem," *IEEE Trans. Automat. Contr.*, 2003.

[4] Busse, F. and How, J., "Real-Time Experimental Demonstration of Precise Decentralized Relative Navigation for Formation Flying Spacecraft," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Monterey, CA, 2002.

[5] Leung, S. and Montenbruck, O., "Real-Time Navigation of Formation-Flying Spacecraft Using Global-Positioning-System Measurements," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 2, 2005.

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

[7] Lawden, D. F., *Optimal Trajectories for Space Navigation*, Butterworth, London, 1963.

[8] Inalhan, G., Tillerson, M., and How, J., "Relative Dynamics and Control of Spacecraft Formations in Eccentric Orbits," *Journal of Guidance, Control and Dynamics*, Vol. 25, No. 1, 2002.

[9] Broucke, R. A., "Solution of the Elliptic Rendezvoud Problem with Time as the Independent Variable," *Journal of Guidance, Control and Dynamics*, Vol. 26, No. 4, 2003, pp. 615–621.

[10] Vallado, D., *Fundamentals of Astrodynamics and Applications*, McGraw-Hill, Space Technology Series.

[11] Mueller., J. and Brito, M., "A Distributed Flight Software Design For Satellite Formation Flying Control," *Proceedings of the AIAA Space 2003 Conference*, No. AIAA 2003-6373, AIAA, Long Beach, CA, 2003.

[12] Mueller, J., "A Multiple-Team Organization for Decentralized Guidance and Control of Formation Flying Spacecraft," *1st AIAA Intelligent Systems Conference*, Chicago, IL, 2004.

[13] Breger, L. and How, J., "GVE-Based Dynamics and Control for Formation Flying Spacecraft," *2nd International Symposium on Formation Flying Missions and Technologies*, 2004.

[14] Carpenter, J. R., Leitner, J., Folta, D., and Burns, R., "Benchmark Problems for Spacecraft Formation Flying Missions," *Proceedings of the AIAA Guidance, Navigation and Control Conference*, No. AIAA 2003-5364, AIAA, Austin, TX, 2003.

[15] Tillerson, M., Inalhan, G., and How, J., "Co-ordination and control of distributed spacecraft systems using convex optimization techniques," *International Journal of Robust Nonlinear Control*, Vol. 12, No. 1, 2002, pp. 207–242.

[16] Breger, L., Ferguson, P., and How, J., "Distributed Control of Formation Flying Spacecraft Built on OA," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Austin, TX, 2003.

[17] Richards, A., Schouwenaars, T., How, J., and Feron, E., "Spacecraft Trajectory Planning with Avoidance Constraints Using Mixed-Integer Linear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 4, 2002.

[18] Tillerson, M. and How, J., "Formation Flying Control in Eccentric Orbits," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Montreal, Canada, 2001.

[19] Hamilton, N., Folta, D., and Carpenter, R., "Formation Flying Satellite Control Around the Sun-Earth L2 Libration Point," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, No. 2002-4528, AIAA, 2002.

[20] Campbell, M., "Collision Monitoring within Satellite Clusters," *IEEE Transactions on Control Systems Technology*, Vol. 13, No. 1, January 2005.