AAS 13-439

DECENTRALIZED MODEL PREDICTIVE CONTROL OF SWARMS OF SPACECRAFT USING SEQUENTIAL CONVEX PROGRAMMING

Daniel Morgan,* Soon-Jo Chung,† and Fred Y. Hadaegh‡

This paper presents a decentralized, model predictive control algorithm for the reconfiguration of swarms of spacecraft composed of hundreds to thousands of agents with limited capabilities. In our prior work, sequential convex programming has been used to determine collision-free, fuel-efficient trajectories for the reconfiguration of spacecraft swarms. This paper uses a model predictive control approach to implement the sequential convex programming algorithm in real-time. By updating the optimal trajectories during the reconfiguration, the model predictive control algorithm results in decentralized computations and communication between neighboring spacecraft only. Additionally, model predictive control reduces the horizon of the convex optimizations, which reduces the run time of the algorithm.

INTRODUCTION

Spacecraft formation flying has been a major area of research over the past decade. Recently, the idea of formation flying has been extended to create swarms of spacecraft,[1] which contain a large number (hundreds to thousands) of femtosatellites (100-gram class spacecraft). Due to their small size, the femtosats have limited sensing, actuation, and computation capabilities, which require the guidance and control algorithms of the swarm to be both fuel and computationally efficient.

J_2-invariant orbits[2] have been shown to maintain the swarm shape and provide collision-free motion for hundreds of orbits. These orbits are very effective at swarm keeping once the swarm is in a desired formation. However, another important requirement for swarm missions is the guidance and control of the swarm reconfiguration. The goal of this paper is to develop a fuel and computationally efficient guidance and control algorithm for the reconfiguration of a swarm of spacecraft located in low Earth orbit (LEO). This algorithm will transfer the spacecraft from one set of J_2-invariant passive relative orbits (PROs) to another. In addition to being fuel and computationally efficient, the algorithm should provide collision-free motion in the highly nonlinear dynamics of relative spacecraft motion in the presence of J_2, which is the dominant perturbation in LEO.

Previous work in spacecraft formation flying[3,4] and multivehicle control research[5,6,7] has developed many multivehicle guidance methods. However, the previous work in formation flying usually

*Graduate Research Assistant, Department of Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, morgan29@illinois.edu, AIAA Student Member
†Assistant Professor, Department of Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, sjchung@illinois.edu, AIAA Senior Member
‡Senior Research Scientist and Technical Fellow, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109, USA, fred.y.hadaegh@jpl.nasa.gov, AIAA Fellow

deals with a small number of spacecraft, a dozen at the most. Additionally, the spacecraft are much larger than femtosats with greater capabilities. The swarm guidance algorithms must be different from previous research because they need to simultaneously address the large number of agents, the modest capabilities of each individual agent, and the complex dynamic environment. Specifically, the large number of spacecraft makes collision avoidance a major challenge and the limited computational capabilities of each agent require that the swarm reconfiguration algorithm is very simple so that it can be run onboard the femtosats in real time.

The reconfiguration of a swarm of spacecraft can be formulated as a nonlinear optimal control problem. Many methods have been developed for solving nonlinear optimal control problems. Due to the complicated nonlinear dynamics of swarms of spacecraft, indirect methods become difficult to use because they require the derivation of the first-order necessary conditions for optimality.[8,9] Therefore, many optimization problems are solved using direct methods, which parameterize the control, and sometimes the state, space reducing the problem to a nonlinear optimization. Pseudospectral methods[10] have been used for trajectory optimization but this method solves a centralized problem which will scale poorly with the number of spacecraft due to the coupling of spacecraft in the collision avoidance constraints. Mixed integer linear programming (MILP) can be used to enforce collision avoidance constraints and has been implemented in real-time[11] as well as used for preplanning trajectories.[12,13] However, these algorithms will also scale poorly as the number of spacecraft increases due to the increase in integer variables caused by the increase in the number of collision constraints.

To find a swarm reconfiguration algorithm that can be run onboard the femtosats, more efficient optimization algorithms are considered. Convex optimization[14] has been used in multi-vehicle trajectory design and it has been shown to efficiently achieve a global optimum. Mattingley et al.[15] used convex optimization to implement a receding horizon controller for a convex problem. Acikmese et al.[16] used convex optimization to find trajectories for a formation reconfiguration with collision avoidance. However, convexifying the collision constraints results in an overly conservative approximation of the collision-free region. In our prior work,[17] sequential convex programming (SCP)[18] was applied to the swarm reconfiguration. SCP uses multiple iterations to ensure that the convex approximations of nonconvex constraints are accurate resulting in more fuel-efficient trajectories. Additionally, the SCP algorithms can be written using freely available software, such as CVX,[19] to solve the convex programming problems at each iteration.

By solving the swarm reconfiguration as an optimization problem, the entire trajectory is generated for each spacecraft at the initial time. In our prior work,[17] it was shown that these trajectories can be computed onboard the femtosats. However, calculating the entire trajectory, with collision avoidance, for each spacecraft at the initial time requires each spacecraft to have strong communication capabilities. In order to relax this assumption, the swarm reconfiguration is formulated as a model predictive control (MPC), or receding horizon control, problem using SCP to solve the optimizations.

MPC has been a major research area in recent years.[20,21] The original MPC problem has been modified to create robust MPC[22,23,24] and fast MPC.[25,26] Additionally, MPC has been used in applications similar to swarm guidance, such as vehicle maneuvering[22] and spacecraft landing.[27] In all of these variations and applications of MPC, the basic idea remains the same. MPC computes the control input by solving a finite-horizon optimization subject to control and state constraints with the current state as the initial state of the optimization. Then, the control input is applied to the system until a new computation is completed giving an updated control input.

The goal of this paper is to develop a model predictive control implementation, which provides collision-free motion for the reconfiguration of swarms of spacecraft and can be implemented on a femtosat with limited computation and communication capabilities. The MPC algorithm presented in this paper will build upon our prior work on J_2-invariant orbits[2] and optimal swarm trajectories.[17] The J_2-invariant conditions from our prior work[2] are used as the boundary conditions for the swarm reconfiguration. The SCP algorithm[17] will be used to compute the optimizations used in the MPC implementation, which results in a fully decentralized reconfiguration algorithm.

The novelty of the MPC implementation using SCP is that it decentralizes the computations and communications required for swarm reconfiguration with collision avoidance. This allows the algorithm to handle hundreds to thousands of spacecraft in real time with calculations performed onboard the femtosats. Additionally, the MPC implementation offers several other advantages. First, the limited horizon of the MPC implementation greatly reduces the size of the SCP problem and, therefore, the run time. Additionally, the limited horizon allows the algorithm to include collision avoidance constraints for only the neighboring spacecraft. This decentralizes the communication requirements of the SCP algorithm. Finally, by running the SCP algorithm multiple times, any differences between the desired and actual trajectories, which can be caused by errors or uncertainties, are accounted for when computing the future trajectories. This provides some robustness to the MPC implementation that is not present when the SCP algorithm is run only once at the initial time.

The paper is organized as follows. First, the swarm reconfiguration is discussed and the optimal control problem is described. Then, the SCP problem is implemented using MPC and the effectiveness of this algorithm is investigated. Finally, the results of simulations are discussed.

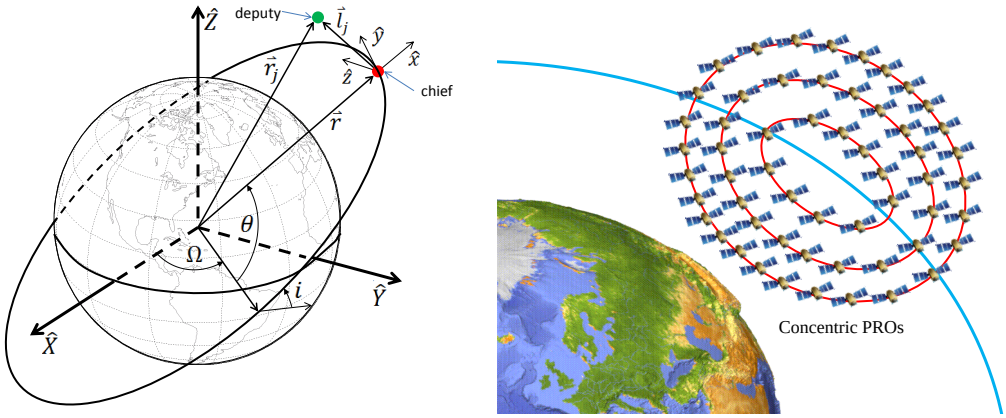
GUIDANCE OF SWARMS OF SPACECRAFT

In this section, the swarm reconfiguration is presented as a continuous, finite horizon optimal control problem. The swarm reconfiguration involves the transfer of hundreds to thousands of spacecraft from one J_2-invariant PRO[2] to another while avoiding collisions between spacecraft and minimizing the total fuel used during the transfer. To properly define the variables and constraints involved in the optimization problem, two coordinate systems must be defined. First, the Earth Centered Inertial (ECI) coordinate system is used to locate the chief spacecraft or a virtual reference point called the chief orbit (see Figure 1a). This coordinate system is inertially fixed and located at the center of the Earth. The $\hat{X}^*$ direction points towards the vernal equinox, the $\hat{Z}$ direction points towards the north pole, and the $\hat{Y}$ direction is perpendicular to the other two and completes the right-handed coordinate system. The second coordinate system is the Local Vertical, Local Horizonal (LVLH) coordinate system. The LVLH frame is centered at the chief spacecraft or chief orbit. Figure 1a shows the LVLH frame with respect to a chief spacecraft. The $\hat{x}$, or radial, direction is always aligned with the position vector and points away from the Earth, the $\hat{z}$, or crosstrack, direction is aligned with the angular momentum vector, and the $\hat{y}$, or alongtrack, direction completes the right-handed coordinate system. The LVLH frame is a rotating frame with a rotation rate of ω_x about the radial axis and ω_z about the crosstrack axis. The relative state of the deputy spacecraft in the LVLH frame is expressed by $\mathbf{x}_j = [\, x_j \quad y_j \quad z_j \quad \dot{x}_j \quad \dot{y}_j \quad \dot{z}_j \,]^T$.

The optimization problem for swarm reconfiguration is written using the LVLH coordinates and dynamics. The equations of motion for spacecraft in the LVLH frame are shown in the Appendix.[28]

The reference orbit should be defined by either a virtual or passive spacecraft so that the equations

*A notation section defining the symbols used in this paper is located after the acknowledgement section.



(a) ECI $(\hat{X}, \hat{Y}, \hat{Z})$ and LVLH Frames $(\hat{x}, \hat{y}, \hat{z})$ (b) Spacecraft Swarm

Figure 1. A visualization of the relative coordinate system and a spacecraft swarm[2]

of motion can be integrated and the reference orbital elements (œ) can be thought of as known values in the optimization problem. The dynamics constraints are given in the Appendix. In addition to the dynamics constraints, the following constraints must be enforced as well.

$$\|\mathbf{u}_j(t)\| \leq U_{\max} \qquad \forall t \in [0, t_f], \qquad j = 1, \ldots, N \tag{1}$$

$$\|C[\mathbf{x}_j(t) - \mathbf{x}_i(t)]\| \geq R_{\text{col}} \qquad \forall t \in [0, t_f], \qquad i > j, \qquad j = 1, \ldots, N-1 \tag{2}$$

$$\mathbf{x}_j(0) = \mathbf{x}_{j,0} \qquad j = 1, \ldots, N \tag{3}$$

$$\mathbf{x}_j(t_f) = \mathbf{x}_{j,f} \qquad j = 1, \ldots, N \tag{4}$$

where $C = [I_{3\times3} \quad 0_{3\times3}]$. Equation (1) represents the limitation on the magnitude of the control vector with $U_{\max}$ being the maximum allowable control magnitude, Eq. (2) is the collision avoidance constraint with R_{col} being the minimum allowable distance between two spacecraft, and Eqs. (3)-(4) are the initial state constraint and the final state constraint, respectively. For the simulations, random positions are chosen for the initial and final states and the velocities are determined using the J_2-invariant conditions developed in our prior work.[2]

The objective of the swarm reconfiguration is to minimize fuel. Therefore, we can define the swarm reconfiguration as follows

Problem 1: Nonconvex Optimization

$$\min_{\mathbf{u}_j, j=1,..,N} \sum_{j=1}^{N} \int_0^{t_f} \|\mathbf{u}_j(t)\| dt \quad \text{subject to} \quad \{(39), (1) - (4)\} \tag{5}$$

It is important to note that the objective function and the constraints of Eqs. (1), (3), and (4) already satisfy the requirements for a convex programming problem. Therefore, only the dynamics, Eq. (39), and the collision avoidance constraints, Eq. (2), need to be converted in order to make Problem 1 (Eq. (5)) convex.

The next step is to express the swarm reconfiguration problem, described in Problem 1, as a convex program. To convexify Problem 1, the dynamic constraints in Eq. (39) are linearized, the state and control variables are discretized, and the collision avoidance constraints in Eq. (2) are

convexified. This convexification process is shown in detail in our prior work.[17] The definitions and derivations of A_d, B_d, and c_d are located in the Appendix. The resulting convex program is shown in Problem 2.

Problem 2: Convexified Optimization

$$\min_{\mathbf{u}_j} \sum_{k=1}^{T-1} \|\mathbf{u}_j[k]\|\Delta t \qquad \forall j = 1, \ldots, N \tag{6}$$

subject to

$$\mathbf{x}_j[k+1] = A_d(\bar{\mathbf{x}}_j, \boldsymbol{\omega})\mathbf{x}_j[k] + B_d\mathbf{u}_j[k] + c_d(\bar{\mathbf{x}}_j, \boldsymbol{\omega}), \quad k = 1, \ldots, T-1, \quad j = 1, \ldots, N \tag{7}$$

$$\|\mathbf{u}_j[k]\| \leq U_{\max} \quad k = 1, \ldots, T-1, \quad j = 1, \ldots, N \tag{8}$$

$$\mathbf{x}_j[1] = \mathbf{x}_{j,0} \quad j = 1, \ldots, N \tag{9}$$

$$\mathbf{x}_j[T] = \mathbf{x}_{j,f} \quad j = 1, \ldots, N \tag{10}$$

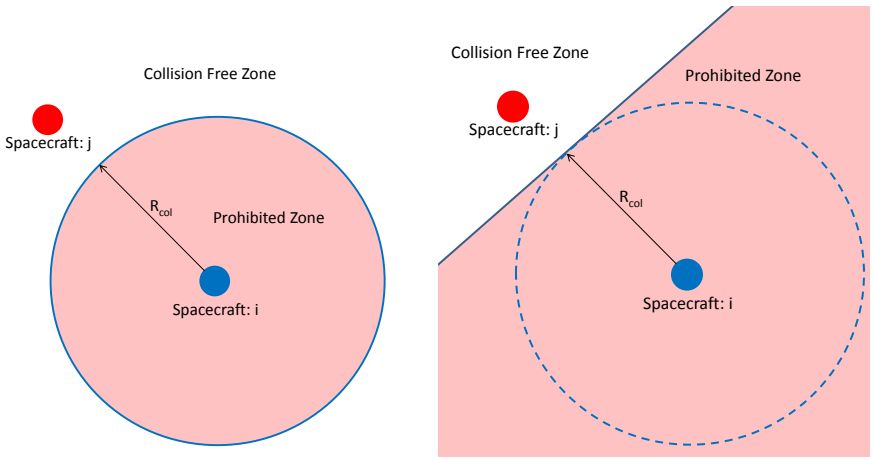
$$[\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k]]^T C^T C[\mathbf{x}_j[k] - \mathbf{x}_i[k]] \geq R_{\text{col}}\|C[\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k]]\| \quad k = 1, \ldots, T, \quad i > j, \quad j = 1, \ldots, N-1 \tag{11}$$

In Problem 2, The collision avoidance constraints are converted to convex constraints. Because the collision avoidance constraints in their original form are the complement of a convex set, the best convex approximations will be affine constraints. In other words, the sphere which defines the forbidden region is approximated by a plane which is tangent to the sphere and perpendicular to the line segment between the two spacecraft. This idea is shown in 2-D using a line and a circle in Figure 2. Figure 2a shows the prohibited zone for the initial collision avoidance constraint and Figure 2b demonstrates the convexification of the constraint. Based on the positions of the spacecraft in the previous iteration of the SCP algorithm, a line (or plane in the 3-D version) is defined tangent to the nonconvex prohibited zone and perpendicular to the line segment connecting the spacecraft. This line defines a half space, which is the convex prohibited zone. As can be seen in Figure 2b, the convex prohibited zone contains the nonconvex prohibited zone so collision avoidance is still guaranteed using this convexification method.

MODEL PREDICTIVE CONTROL

Problem Formulation

In this section, the convex program described in Problem 2 is converted to a MPC problem. In order to describe the MPC algorithm, Problem 3 and Problem 4 are defined. Problem 3 is defined so that the horizon for the optimization does not reach the terminal time for the reconfiguration. For this reason, the terminal constraint, Eq. (10), in Problem 2 is not enforced in Problem 3. Instead, a terminal cost ($h_j(\mathbf{x}[k], k)$) is added to the objective to estimate the cost of completing the reconfiguration from the state and time at the end of the optimization horizon. Problem 3 is used in the MPC algorithm when the horizon of the optimization does not reach the terminal time of the reconfiguration. Problem 4 is very similar to Problem 2 with the only difference being the starting time. Problem 4 is used in the MPC algorithm when the horizon of the optimization goes beyond the terminal time of the reconfiguration. In both Problem 3 and Problem 4, the spacecraft are assumed



(a) Nonconvex prohibited zone (b) Convex approximation of prohibited zone

Figure 2. Convexification of the 2-D collision avoidance constraint[17]

to have limited communication range. Therefore, they can only communicate with the other spacecraft that are near them. This determines which pairs of spacecraft will have collision constraints. Problem 3 and Problem 4 are expressed as follows.

Problem 3: Convex Optimization with Terminal Cost for $k_0 + T_H < T$

$$\min_{\mathbf{u}_j} \sum_{k=k_0}^{k_0+T_H-1} \|\mathbf{u}_j[k]\|\Delta t + h_j\left(\mathbf{x}_j[k_0+T_H], k_0+T_H\right) \qquad \forall j = 1,\ldots,N \tag{12}$$

subject to

$$\mathbf{x}_j[k+1] = A_d(\bar{\mathbf{x}}_j, \mathbf{oe})\mathbf{x}_j[k] + B_d\mathbf{u}_j[k] + c_d(\bar{\mathbf{x}}_j, \mathbf{oe}), \qquad k = k_0,\ldots,k_0+T_H-1, \qquad j = 1,\ldots,N \tag{13}$$

$$(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k])^T C^T C(\mathbf{x}_j[k] - \bar{\mathbf{x}}_i[k]) \geq R_{\text{col}}\|C(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k])\| \tag{14}$$
$$k = k_0,\ldots,k_0+T_H, \quad \{i,j\} : i \in \mathcal{N}_j, \quad \mathcal{N}_j = \{i | i > j, \|\mathbf{x}_j[k_0] - \mathbf{x}_i[k_0]\| \leq R_{\text{comm}}\}$$

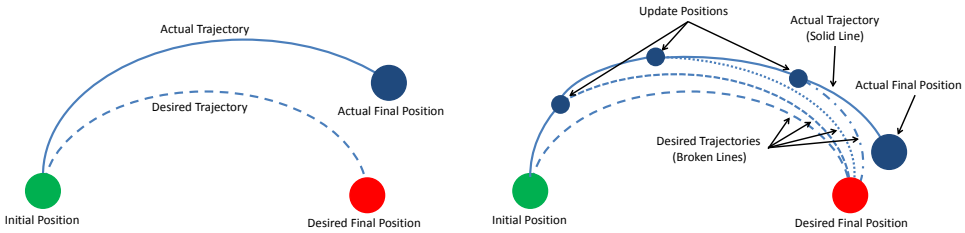
$$\|\mathbf{u}_j[k]\| \leq U_{\max} \qquad k = k_0,\ldots,k_0+T_H-1, \qquad j = 1,\ldots,N \tag{15}$$
$$\mathbf{x}_j[k_0] = \mathbf{x}_{j,\text{MPC}} \qquad j = 1,\ldots,N \tag{16}$$

Problem 4: Convex Optimization with Terminal Constraint for $T - T_H \leq k_0 < T$

$$\min_{\mathbf{u}_j} \sum_{k=k_0}^{T-1} \|\mathbf{u}_j[k]\|\Delta t \qquad \forall j = 1,\ldots,N \tag{17}$$

subject to (16) and

$$\mathbf{x}_j[k+1] = A_d(\bar{\mathbf{x}}_j, \mathbf{oe})\mathbf{x}_j[k] + B_d\mathbf{u}_j[k] + c_d(\bar{\mathbf{x}}_j, \mathbf{oe}), \qquad k = k_0,\ldots,T-1, \qquad j = 1,\ldots,N \tag{18}$$



(a) Trajectory from running the SCP algorithm at the initial time only

(b) Trajectory from running the MPC implementation of the SCP algorithm

Figure 3. Convexification of the 2-D collision avoidance constraint

$$(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k])^T C^T C(\mathbf{x}_j[k] - \bar{\mathbf{x}}_i[k]) \geq R_{\text{col}} \|C(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k])\| \tag{19}$$
$$k = k_0, \ldots, T, \quad \{i, j\} : i \in \mathcal{N}_j, \quad \mathcal{N}_j = \{i | i > j, \|\mathbf{x}_j[k_0] - \mathbf{x}_i[k_0]\| \leq R_{\text{comm}}\}$$

$$\|\mathbf{u}_j[k]\| \leq U_{\max} \qquad k = k_0, \ldots, T-1, \qquad j = 1, \ldots, N \tag{20}$$

$$\mathbf{x}_j[T] = \mathbf{x}_{j,f} \qquad j = 1, \ldots, N \tag{21}$$

The MPC implementation of the SCP algorithm (MPC-SCP) is performed by reducing the horizon of the SCP problem and then solving this problem repeatedly throughout the reconfiguration. Initially, the SCP algorithm[17] is run for the optimal trajectory up to a finite horizon (T_H). As the spacecraft approaches this horizon in real time, Problem 3 is solved using SCP for the current time step (k_0) and position ($\mathbf{x}_{j,\text{MPC}}$) up to the new horizon ($k_0 + T_H$). It is important to note that k_0 is the current time step at the beginning of each MPC iteration and increases with time. $\mathbf{x}_{j,\text{MPC}}$ is the real-time position and velocity of the spacecraft when the MPC algorithm is run. This value represents the initial condition of the MPC algorithm. Once k_0 exceeds $T - T_H$, Problem 4 is solved instead of Problem 3. This process is repeated until the spacecraft reaches the desired position ($\mathbf{x}_{j,f}$) at the final time step (T).

The result of the MPC implementation is a fully decentralized reconfiguration algorithm with improved computation times as well as better robustness when sensor and actuator errors are included. The decentralization of the swarm reconfiguration algorithm greatly reduces the communication and computation requirements of the femtosats. Additionally, the increased robustness properties of this algorithm will reduce the fuel requirements for the femtosats. The benefits of the MPC implementation with respect to robustness and fuel efficiency are shown in Figure 3. Figure 3a shows how an initial actuator or sensor error can cause the actual final position (blue circle) to have a significant error with respect to the desired final position (red circle) if the SCP algorithm is only run once. However, the MPC implementation in Figure 3b can reduce this error by updating the desired trajectory based on the actual position and velocity at various points (small blue circles) throughout the reconfiguration. In addition to reducing final position errors, the MPC implementation reduces the computation, communication, and fuel requirements, which is especially important for femtosats due to their very limited volume and mass.

Stability

The stability of the MPC-SCP algorithm is dependent on the terminal cost function ($h_j\left(\mathbf{x}_j[k], k\right)$) in Eq. (12) of Problem 3. In order to ensure the stability of the MPC algorithm, the terminal cost function ($h_j\left(\mathbf{x}_j[k], k\right)$) is defined as the solution to the following optimization problem.

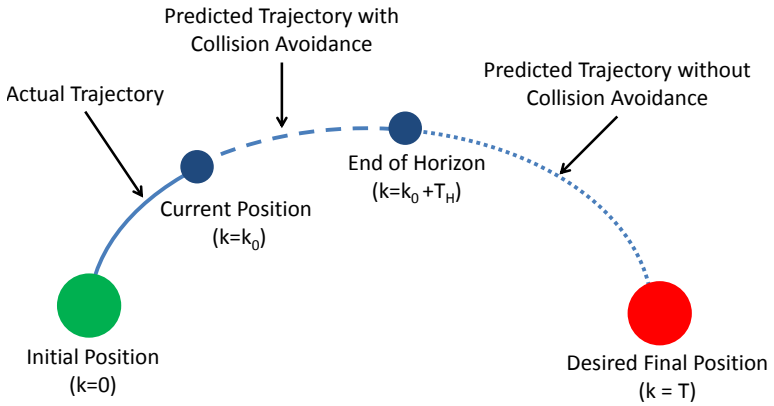


Figure 4. Illustration of the optimization horizon used in the MPC algorithms

Problem 5: Convex Terminal Cost Function

$$\text{Minimize (17) subject to } \{(16),(18),(20),(21)\}$$

This terminal cost function evaluates the fuel required to go from the position and time at the end of the horizon to the terminal position at the terminal time without considering collision avoidance constraints during this part of the trajectory. There are two reasons not to enforce the collision avoidance constraints when calculating the terminal cost function. First, the collision avoidance constraints add complexity to the problem so removing them greatly reduces the time required for the computation. Second, the spacecraft can only communicate with other spacecraft within a certain distance of them.

Substituting the terminal cost function that results from Problem 5 into Problem 3 results in the following optimization problem.

Problem 6: Stable Convex Optimization for $k_0 + T_H < T$

$$\text{Minimize (17) subject to } \{(14),(16),(18),(20),(21)\}$$

The concept of Problem 6 is shown in Figure 4. This figure shows the various stages of the MPC algorithm. The first stage is shown by the solid line in Figure 4. This represents the actual trajectory that the spacecraft has traversed and it occurs between $k = 1$ and $k = k_0$. $k = k_0$ represents the current time, and the initial time in the optimization. The next stage occurs between $k = k_0$ and $k = k_0 + T_H$ and is shown as a dashed line in the figure. This represents the predicted trajectory and collision avoidance is considered during this time period. The final stage is illustrated by the dotted line and extends from $k = k_0 + T_H$ and $k = T$. During this time, the predicted trajectory does not take into account collision avoidance. It is important to note that if the second stage (dashed line) extends beyond the final time, the final stage does not exist and Problem 4 should be used instead of Problem 6.

Stable MPC-SCP uses Problem 4 and Problem 6 as the optimization problems. This algorithm uses the solution to Problem 4 if $k_0 \geq T - T_H$ and the solution to Problem 6 if $k_0 < T - T_H$. In either case, the final state is fixed from Eq. (21) in Problem 4 and Problem 6. Therefore the

trajectory is guaranteed to converge to the desired final state as long as the optimizations described in Problems 4 and 6 are feasible. The feasibility of the optimizations is discussed in the following subsection.

Feasibility

In order for the trajectories resulting from using Stable MPC-SCP to converge, the optimizations must be feasible. Infeasibility of the optimization can result for two reasons: The collision avoidance constraints cannot all be satisfied or the terminal constraint cannot be satisfied without violating the limit on the velocity and/or control vectors. The collision avoidance infeasibility arises because collision avoidance is only considered up to the horizon of the optimization and other spacecraft can only be detected if they are within the communication radius (R_{comm}). Therefore, collisions that occur after the optimization horizon or with spacecraft outside of the communication radius are not considered until a later time step. For this reason, several conditions are introduced to ensure that collision avoidance is guaranteed.

In order to guarantee feasibility, an artificial constraint is imposed on the problem in order to bound the distance that each spacecraft can move during each time step. This condition is written as follows.

$$\|D\mathbf{x}_j[k]\| \leq V_{\max} \qquad k = k_0, \ldots, T, \qquad j = 1, \ldots, N \tag{22}$$

where $D = [0_{3\times3} \quad I_{3\times3}]$. Adding this constraint to Problem 6 and Problem 4 yields Problem 7 and Problem 8, respectively.

Problem 7: Feasible Convex Optimization for $k_0 + T_H < T$

Minimize (17) subject to {(14),(16),(18),(20),(21),(22)}

Problem 8: Feasible Convex Optimization for $T - T_H \leq k_0 < T$

Minimize (17) subject to {(16),(18),(19),(20),(21),(22)}

Feasible MPC-SCP uses Problem 7 and Problem 8 as the optimization problems. This method uses the solution to Problem 7 if $k_0 < T - T_H$ and the solution to Problem 8 if $k_0 \geq T - T_H$. Assuming that the original optimization problem described by Problem 2 is feasible, i.e. the initial and terminal constraints can be satisfied without violating the collision avoidance constraints, the following conditions ensure that the MPC-SCP algorithm is feasible :

Proposition 1: *Detectable Collisions*
All spacecraft that can cause collisions within the current horizon are able to be detected if

$$R_{\text{comm}} \geq \max\{2V_{\max}T_H\Delta t, R_{\text{col}} + 2V_{\max}\Delta t\} \tag{23}$$

Proof: This first term on the right hand side guarantees that any spacecraft that could potentially cause a collision before the end of the MPC horizon is detected and therefore considered in the optimization. The length of the horizon is the number of time steps (T_H) multiplied by the length of each time step (Δt). Additionally, the maximum relative velocity between two spacecraft is $2V_{\max}$. Therefore, the maximum change in the relative distance between two spacecraft is given

by the first term on the right hand side of Eq. (23). Any pair of spacecraft closer than this distance can potentially collide before the end of the MPC horizon. To ensure that all of these collisions are detected, the communication radius (R_{comm}) must be at least as big as this distance. This establishes $R_{\text{comm}} \geq 2V_{\max}T_H\Delta t$.

This second term on the right hand side ensures that the initial position constraints for each optimization do not violate the collision avoidance constraints. To ensure this, every collision must be detected at least one time step before the collision occurs. As mentioned above, the maximum relative velocity between two spacecraft is $2V_{\max}$. This means that the change in the relative distance between two spacecraft in one time step is $2V_{\max}\Delta t$. This distance must be less than the difference between the communication radius (R_{comm}) and the collision radius (R_{col}). This establishes $R_{\text{comm}} \geq R_{\text{col}} + 2V_{\max}\Delta t$ and Eq. (23). □

Proposition 2: *Computational Feasibility*
The new control sequence can be computed before the previous horizon is reached if

$$t_{\text{run}} \leq T_H\Delta t \tag{24}$$

Proof: Since collision avoidance is not enforced after the end of the MPC horizon, a new control sequence must be computed before the current control sequence reaches the end of the horizon. Otherwise, the control sequence will not necessarily avoid collisions. Therefore, the computation time of each step of the MPC algorithm (t_{run}) must be less than the length of the MPC horizon ($T_H\Delta t$). This results in Eq. (24). □

Propositions 1-2 ensure the optimizations in Feasible MPC-SCP have solutions. Since the optimizations performed by this algorithm are feasible, the collision avoidance constraints are satisfied and there are no collisions at the discrete time steps. However, there is still a possibility that collisions occur in between time steps when the collision avoidance constraints are not enforced. The following theorem addresses this issue.

Theorem 1: *Collision Avoidance between Time Steps*
If two spacecraft are collision free during two consecutive time steps k and $k+1$ and Eqs. (25)–(26) are satisfied, then the two spacecraft are collision free in the interval $t \in [k\Delta t, (k+1)\Delta t]$.

$$V_{\max} < \frac{R_{\text{col}}}{\Delta t} \tag{25}$$

$$R_{\text{col}} \geq \begin{cases} \sqrt{(\bar{R}_{\text{col}} + \frac{a^*\Delta t^2}{4})^2 + (V_{\max}\Delta t)^2 - \frac{(a^*\Delta t^2)^2}{4}} & \text{if} \quad a^* < \min\{\frac{2V_{\max}}{\Delta t}, a_{\max}\} \\ \sqrt{(\bar{R}_{\text{col}} + \frac{a_{\max}\Delta t^2}{4})^2 + (V_{\max}\Delta t)^2 - \frac{(a_{\max}\Delta t^2)^2}{4}} & \text{else if} \quad a_{\max} < \frac{2V_{\max}}{\Delta t} \\ \bar{R}_{\text{col}} + \frac{V_{\max}\Delta t}{2} & \text{else} \end{cases} \tag{26}$$

where

$$a^* = \frac{2}{\sqrt{3}}\sqrt{\frac{R_{\text{col}}^2}{\Delta t^4} - \frac{V_{\max}^2}{\Delta t^2}} \tag{27}$$

Proof: The Feasible MPC-SCP algorithm guarantees that the trajectories are collision free at the discrete time steps. However, the trajectories must also be collision free in between time steps in order to guarantee collision-free trajectories. The first step to ensuring that collisions do not occur in between time steps is to establish a condition which prevents two spacecraft from passing through each other. Two spacecraft can move by a relative distance of $2V_{\max}\Delta t$ in one time step. This

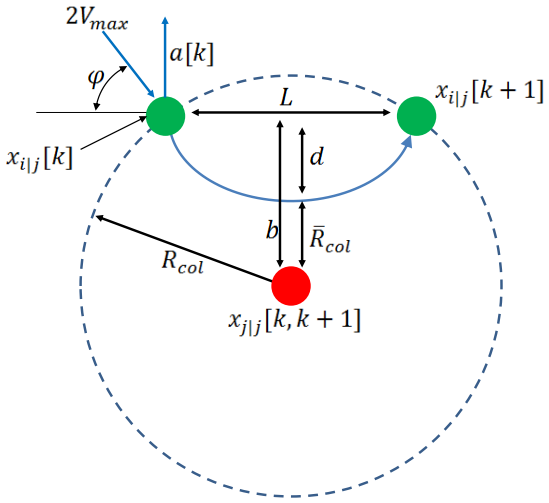


Figure 5. Illustration of the worst case scenario for collisions in between time steps

distance must be less than twice the collision radius (R_{col}) to prevent the spacecraft from passing through each other. This establishes Eq. (25).

Now that the spacecraft cannot pass through each other, the minimum possible relative distance between the spacecraft is established for a given ($V_{\max}, a_{\max}$) where $V_{\max}$ is the maximum allowable velocity and $a_{\max}$ is the maximum allowable acceleration, which includes acceleration due to both the control and the dynamics. The discretization method uses a constant acceleration in between time steps. Consider this scenario from the reference frame centered at spacecraft j as shown in Figure 5. In this figure, the subscript $i|j$ denotes the location of spacecraft i with respect to j where $i \in \mathcal{N}_j$. In the worst case scenario, the distance between the two spacecraft is R_{col} at both time steps and both the relative velocity and acceleration vectors are in the plane defined by spacecraft j and the initial and final positions of spacecraft i. This scenario is depicted in Figure 5 and the distances in Eqs. (28)–(31) are defined in this figure.

Since the acceleration vector is constant, the minimum distance ($\bar{R}_{\text{col}}$) occurs when the problem is symmetric as shown in Figure 5.

$$L = 2V_{\max}\cos\phi\Delta t \tag{28}$$

$$b = \sqrt{R_{\text{col}}^2 - (\frac{L}{2})^2} \tag{29}$$

$$d = V_{\max}\sin\phi\Delta t - a\frac{\Delta t^2}{4} \tag{30}$$

$$\bar{R}_{\text{col}} = b - d \tag{31}$$

where L is length of the segment connecting the position of spacecraft i relative to spacecraft j at time steps k and $k+1$, b is the minimum distance between this segment and spacecraft j, d is the distance between the line segment measured by L and the point of closest approach of spacecraft i relative to spacecraft j, and ϕ is the angle between the initial velocity vector and the line segment

measured by L. Since the closest distance occurs at $\frac{\Delta t}{2}$ due to symmetry, the velocity in the direction of the acceleration vector must be zero at this time.

$$2V_{\max} \sin \phi = a\Delta t \tag{32}$$

This equation holds for $a \leq \frac{2V_{\max}}{\Delta t}$. Substituting Eq. (32) into Eq. (30) results in

$$d = \frac{a\Delta t^2}{4} \tag{33}$$

Additionally, solving Eq. (32) for $\sin \phi$ and substituting it into the identity $\cos^2 \phi = 1 - \sin^2 \phi$ yields

$$\cos^2 \phi = 1 - \frac{a^2 \Delta t^2}{4V_{\max}^2} \tag{34}$$

Combining Eqs. (28), (29), (31), (33), and (34) results in

$$\bar{R}_{\text{col}} = \sqrt{R_{\text{col}}^2 - (V_{\max}\Delta t)^2 + \frac{a^2\Delta t^4}{4}} - \frac{a\Delta t^2}{4} \tag{35}$$

To find the closest approach, Eq. (35) is minimized with respect to a. Taking the first and second derivatives yields

$$\frac{d\bar{R}_{\text{col}}}{da} = (R_{\text{col}}^2 - (V_{\max}\Delta t)^2 + \frac{a^2\Delta t^4}{4})^{-\frac{1}{2}}(\frac{a\Delta t^4}{4}) - \frac{\Delta t^2}{4} \tag{36}$$

$$\frac{d^2\bar{R}_{\text{col}}}{da^2} = -(R_{\text{col}}^2 - (V_{\max}\Delta t)^2 + \frac{a^2\Delta t^4}{4})^{-\frac{3}{2}}(\frac{a\Delta t^4}{4})^2 + (\frac{\Delta t^4}{4})(R_{\text{col}}^2 - (V_{\max}\Delta t)^2 + \frac{a^2\Delta t^4}{4})^{-\frac{1}{2}} \tag{37}$$

Setting Eq. (36) equal to zero establishes Eq. (27) and rearranging Eq. (37) shows that $\frac{d^2\bar{R}_{\text{col}}}{da^2} > 0$ if $V_{\max} < \frac{R_{\text{col}}}{\Delta t}$. This condition has already been established in Eq. (25). Therefore, a^* minimizes $\bar{R}_{\text{col}}$ so long as $a^* \leq \min\{a_{\max}, \frac{2V_{\max}}{\Delta t}\}$. In fact, the minimizing feasible a is the minimum of a^*, $a_{\max}$ and $\frac{2V_{\max}}{\Delta t}$. Substituting these three values into Eq. (35) and solving for $\bar{R}_{\text{col}}$ establishes the three conditions in Eq. (26). □

In addition to infeasibility caused by collision avoidance constraints, infeasibility can also arise due to the constraints on maximum velocity and control magnitudes. Once again, assume that the original optimization described by Problem 2 is feasible. It is possible that the MPC-SCP optimizations are infeasible even when the original problem is feasible. This occurs because the spacecraft have a limited communication radius in the MPC formulation and, therefore, cannot detect collisions occurring after the MPC horizon. This infeasibility is much more likely to occur in situations where the maximum velocity and/or control are achieved in the original problem. Therefore, the swarm reconfiguration should be strictly feasible with respect to the maximum velocity and control constraints when solved using Problem 2. Additionally, $V_{\max}$ is an artificial constraint that was introduced to guarantee collision avoidance. Therefore, it is an optimization parameter rather than a value determined by the problem. To reduce the likelihood that the maximum velocity causes

infeasibility, $V_{\max}$ should be chosen to be as large as possible while still satisfying Propositions 1-2 and Theorem 1.

These methods will greatly reduce the probability that infeasibility will occur but do not guarantee that it cannot happen. If the optimization is infeasible due to maximum velocity or control constraints, the final time can be extended to make the optimization feasible. This can be done in a few different ways depending on the swarm reconfiguration. First, the final time can be extended by a fraction of an orbit without adjusting the final position. This is desirable if the elapsed time of the reconfiguration is critical. The drawback to this method is that it is not as fuel efficient as other options since most of the spacecraft will have to adjust their trajectories for the new final time. The next option is to extend the final time and adjust the final position to the position on the J_2-invariant orbit where the spacecraft would have been if the optimization was completed as planned. This saves both time and fuel but requires an extra calculation for the updated terminal positions. Finally, the final time can be extended by a full orbit. This option is fuel efficient compared to the first one but requires more time. Any of these options will result in a feasible problem but the mission will dictate which one is the most practical.

Remark 1:

When uncertainties are included in the system, forcing the spacecraft to reach a terminal position is unrealistic. Therefore, the terminal constraint can be relaxed to the following condition.

$$\|\mathbf{x}_j[T] - \mathbf{x}_{j,f}\| \leq \delta \tag{38}$$

where $\delta > 0$ is the radius of a ball around $\mathbf{x}_{j,f}$ which must contain the terminal position of spacecraft j.

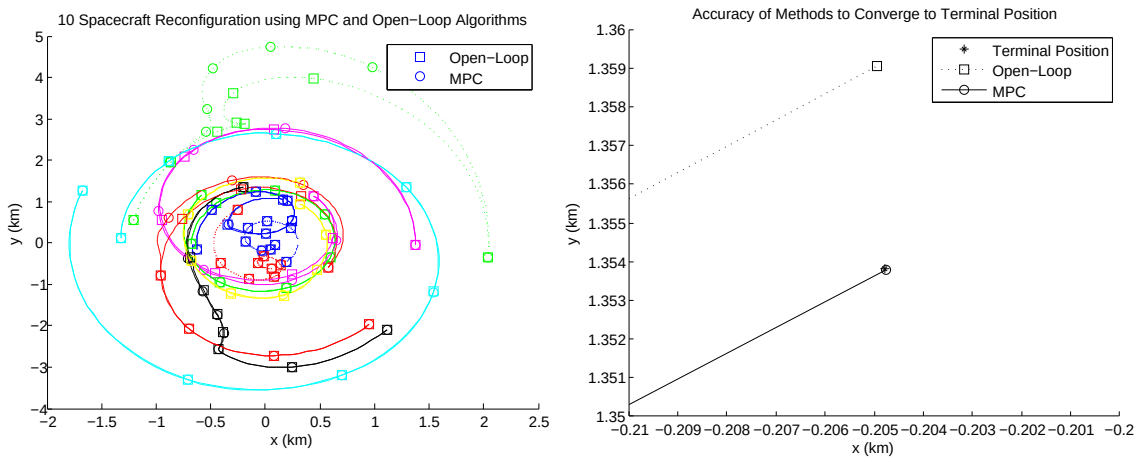
NUMERICAL SIMULATIONS

In this section, simulations of the swarm reconfiguration are presented using the MPC algorithm developed in this paper. The MPC algorithm is compared to an open-loop optimization and the fuel efficiency and accuracy of the trajectories are compared. A formation reconfiguration with 10 spacecraft is solved using Feasible MPC-SCP and the results are compared to an open-loop optimization.

All of the simulations are run with a reference orbit having the following initial orbital elements: 6878 km semimajor axis, 0 eccentricity, 45 deg inclination, 60 deg right ascension , 0 deg argument of perigee, and 0 deg true anomaly. Additionally, the length of the transfer, t_f, is 5677 s, or one orbit and the SCP algorithm is considered to be converged for errors less than $\epsilon = 10^{-3}$. In all the simulations, the initial and terminal conditions, $\mathbf{x}_{j,0}$ and $\mathbf{x}_{j,f}$, respectively, are determined by randomly generating the positions and then applying the J_2-invariant conditions from out prior work[2] to determine the desired velocities. All of the convex optimizations were performed using CVX.[19]

The simulation results for the 10 spacecraft formation reconfiguration are shown in Figure 6 and Table 1. Both MPC and the open-loop algorithms were run for collision radii (R_{col}) of 150 m, a time step (Δt) of 60 s and an optimization horizon (T_H) of 3 time steps. The $x - y$ projection of the trajectories resulting from these algorithms are shown in Figure 6.

Figure 6a shows the reconfiguration trajectories for 10 spacecraft using both MPC and open-loop algorithms. In some cases, such as the green dashed trajectory, the MPC and open-loop methods



(a) View of a 10 spacecraft reconfiguration

(b) Enlargement of the terminal position for one spacecraft

Figure 6. x-y projection of the reconfiguration of 10 spacecraft using MPC and open-loop algorithms

produce different results. This is due to the linearization and discretization errors that arise from converting the optimization to convex form. In the open-loop case, these errors are not accounted for and the resulting trajectory may not reach the terminal state. On the other hand, the MPC algorithm uses real-time positions to calculate the remaining trajectory so the terminal error is much smaller. These errors are shown for a single spacecraft in Figure 6b. The MPC algorithm (solid line) reaches a terminal position (circle) that is less than 50 mm from the desired terminal state (star). This is over two orders of magnitude better than the open-loop trajectory (dotted line), which reaches a terminal position (square) over 5 m from the desired state. The terminal errors for the other spacecraft, along with fuel usage are shown in Table 1.

Table 1 shows the terminal position error and fuel usage for each spacecraft for both the MPC and open-loop trajectories. On average, the terminal error decreases by almost two orders of magnitude when the MPC algorithm is used instead of the open-loop optimization. However, the fuel usage required to correct for these errors using MPC is about 7% more than the fuel usage of the open-loop method. It is important to note, that the increase in fuel consumption is largely due to spacecraft 2. The reason for this has to do with the decentralized communication in the MPC algorithm. In the MPC algorithm, spacecraft only consider collisions up to the end of the horizon, which is 3 time steps in this simulation. In Figure 6, spacecraft 2 (solid red line) performs a large maneuver when the MPC algorithm is used. This maneuver occurs because a future collision is detected and must be avoided in less than three time steps. This maneuver requires a large amount of fuel and does not occur in the open-loop case because the communication is centralized and all collisions are known at the initial time. Additionally, the MPC algorithm satisfies the collision avoidance constraints when nonlinear, continuous dynamics are used to simulate the motion. This is not necessarily true for the open-loop case. While the open-loop trajectories are collision free, the actual trajectories that result from simulating the open-loop control do not necessarily satisfy the collision avoidance requirements. Overall, the MPC algorithm greatly improves the accuracy of the terminal state, decentralizes the communication of the swarm, and guarantees collision avoidance with the only disadvantage being a small increase in fuel consumption.

Table 1. Simulation results for the reconfiguration of a 10 spacecraft formation using decentralized SCP algorithms

		Algorithm Performance	
Algorithm	Spacecraft	Fuel Cost (m/s)	Terminal Error (m)
Feasible MPC-SCP	1	1.451	0.014
	2	2.067	0.013
	3	0.415	0.022
	4	1.146	0.027
	5	3.171	0.050
	6	4.047	0.024
	7	3.635	0.041
	8	2.648	0.036
	9	1.081	0.004
	10	2.754	0.089
	average	2.242	0.032
Open-Loop Optimization[17]	1	1.4491	0.818
	2	0.893	3.577
	3	0.410	0.608
	4	1.048	2.294
	5	3.149	2.741
	6	4.009	3.607
	7	3.602	5.249
	8	2.633	3.376
	9	1.076	0.163
	10	2.736	0.494
	average	2.100	2.293

The previous simulation demonstrated the advantages of the MPC algorithm compared to the open-loop method. In the next simulation, the conditions in Propositions 1-2 and Theorem 1 are enforced to ensure that the algorithm is feasible and avoids collisions between time steps. This simulation has the same parameters as the previous one with the following exceptions: $V_{\max} = 0.005$ km/s, $U_{\max} = 0.001$ km/s^2, $R_{\text{comm}} = 2$ km, $T_H = 12$, and $\Delta t = 15$ s. The results of this simulation are shown in Figure 7.

In this simulation, the average fuel use is 2.164 m/s and the average terminal error is 0.003 m. Both of these values are improvements compared to the previous simulation, which used larger time steps. With a smaller time step, the spacecraft can execute maneuvers more often. Therefore, it makes sense that both the accuracy and fuel efficiency are improved. The decrease in time step size does increase the number of variables in the optimizations, which increases the run time. However, even with the small time steps, the average computation time is about 12 s, which is on the order of one time step and much shorter than the horizon length. Therefore, the simulation is computationally feasible and produces better results than the simulation with larger time steps and no guarantees about feasibility.

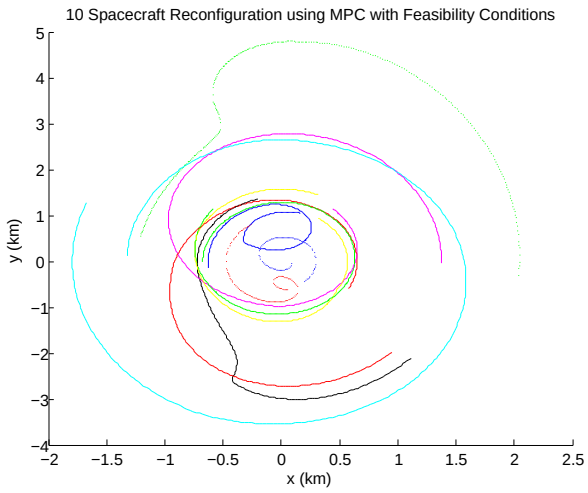


Figure 7. x-y projection of the reconfiguration of 10 spacecraft using MPC with 15 s time steps

CONCLUSION

In this paper, the swarm reconfiguration was solved using a MPC algorithm with a SCP optimization. The swarm reconfiguration was first written as a convex optimization problem and then the problem was formulated with a receding horizon and MPC was applied. Using MPC decreased the size of the optimizations that needed be solved, which allowed smaller time steps in the optimizations. By using smaller time steps and shorter horizons, the MPC algorithm restricted the distance each spacecraft could travel during one optimization. This allowed us to relax the communication requirements on each spacecraft by considering communication between two spacecraft only if they were within a certain distance from one another.

In order to ensure that the trajectories resulting from the MPC optimizations converged to the terminal states, the terminal cost function was converted to a convex optimization problem with a terminal constraint. This constraint ensured that if the optimization was feasible, it would satisfy the terminal conditions. Also, an upper bound on the magnitude of the velocity was introduced so that two propositions could be developed to ensure that each of the spacecraft converged to their desired terminal positions and to show that the receding horizon optimizations had a solution. Additionally, a theorem was developed to guarantee that the spacecraft do not collide in between time steps.

These feasibility conditions were then applied to a randomly distributed swarm and the MPC algorithm was used to compute the optimal trajectories. These results performed well compared to the trajectories which result from solving a single optimization at the initial time. The MPC algorithm drove the spacecraft to within several mm of the desired terminal state despite the linearization and discretization errors of the optimization. On the other hand, the single optimization trajectory missed the desired terminal state by an average of 2 m. Additionally, the time required to run each optimization of the MPC algorithm was much less than the time required to solve for the entire trajectory at the initial time.

Swarms of spacecraft can be an extremely useful tool for interferometry and distributed sensing but in order for these missions to become practical, fuel and computationally efficient guidance and control algorithms must be developed. The fuel and computationally efficient MPC algorithm de-

veloped in this paper is a necessary step towards this goal. Due to the orders of magnitude increase in the number of spacecraft and decrease in the size of the spacecraft, fuel,communication, and computation requirements become very restrictive. The MPC algorithm developed in this paper enables swarms of spacecraft to change their formation using minimal fuel and computational resources.

ACKNOWLEDGMENTS

This research was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. ©2013 California Institute of Technology. This work was supported by a NASA Office of the Chief Technologists Space Technology Research Fellowship. Government sponsorship acknowledged.

NOTATION

J_2	second harmonic coefficient of earth
N	number of spacecraft
R_{col}	minimum distance between spacecraft to avoid a collision in the optimization
$\bar{R}_{\text{col}}$	minimum distance between spacecraft to avoid a collision in reality
R_{comm}	maximum distance a spacecraft can communicate ($R_{\text{comm}} > R_{\text{col}}$)
T	total number of time steps
T_H	number of time steps in the model predictive control horizon
$U_{\max}$	maximum allowable magnitude of the control vector
$V_{\max}$	maximum allowable magnitude of the relative velocity vector
$(\hat{X}, \hat{Y}, \hat{Z})$	Earth centered inertial coordinate system
a	magnitude of the acceleration vector
a^*	acceleration magnitude which minimizes the distance between two spacecraft
$a_{\max}$	maximum possible acceleration of a spacecraft
h	angular momentum
$h_j(\mathbf{x}[k], k])$	cost to transfer a spacecraft from $\mathbf{x}[k]$ at time k to $\mathbf{x}_f$ at T
i	orbit inclination
k	time step k
k_0	time step at the start of the model predictive control horizon
k_{J2}	$\frac{3}{2} J_2 \mu R_e^2, 2.633 \times 10^{10}$ [km^5/s^2]
$\mathbf{œ}$	orbital element vector
r	geocentric distance
r_{jZ}	distance from satellite to equator
t	time
t_f	final time ($t_f = T\Delta t$)
t_{run}	time required to compute the optimization
Δt	length of time step
$\mathbf{u}$	control vector in local vertical, local horizontal frame
v_x	radial velocity
$\mathbf{x}$	state vector in local vertical, local horizontal frame
$\mathbf{x}_0$	state vector at initial time
$\mathbf{x}_f$	state vector at final time
$\mathbf{x}_j$	state vector of spacecraft j
$\mathbf{x}_{\text{MPC}}$	state vector at the start of the model predictive control horizon

$\bar{\mathbf{x}}$	nominal state vector
(x, y, z)	coordinate values in the local vertical, local horizontal coordinate system
$(\hat{x}, \hat{y}, \hat{z})$	unit vectors of the local vertical, local horizontal coordinate system
Ω	right ascension of the ascending node
α_x	angular acceleration of coordinate system about x-axis
α_z	angular acceleration of coordinate system about z-axis
θ	argument of latitude
ω_x	rotation rate of coordinate system about x-axis
ω_y	rotation rate of coordinate system about y-axis
ω_z	rotation rate of coordinate system about z-axis
$\|\cdot\|$	2-norm of a vector

APPENDIX: LINEARIZATION AND DISCRETIZATION OF RELATIVE DYNAMICS

The equations of motion for spacecraft in the LVLH frame ($\boldsymbol{\ell}_j = (x_j, y_j, z_j)^T$) are[28]

$$\ddot{\boldsymbol{\ell}}_j = -2\mathbf{S}(\boldsymbol{\omega})\dot{\boldsymbol{\ell}}_j - \mathbf{g}(\boldsymbol{\ell}_j, \mathbf{oe}) + \mathbf{u}_j \tag{39}$$

The translational dynamics of spacecraft in the LVLH frame is described by Eq. (39) with

$$\mathbf{g}(\boldsymbol{\ell}, \mathbf{oe}) = \begin{bmatrix} \eta_j^2 - \omega_z^2 & -\alpha_z & \omega_x\omega_z \\ \alpha_z & \eta_j^2 - \omega_z^2 - \omega_x^2 & -\alpha_x \\ \omega_x\omega_z & \alpha_x & \eta_j^2 - \omega_x^2 \end{bmatrix} \boldsymbol{\ell} + (\zeta_j - \zeta) \begin{pmatrix} \sin i \sin\theta \\ \sin i \cos\theta \\ \cos i \end{pmatrix} + \begin{pmatrix} r(\eta_j^2 - \eta^2) \\ 0 \\ 0 \end{pmatrix} \tag{40}$$

where

$$\begin{aligned} \zeta &= \frac{2k_{J2}\sin i \sin\theta}{r^4} & \zeta_j &= \frac{2k_{J2}r_{jZ}}{r_j^5} \\ \eta^2 &= \frac{\mu}{r^3} + \frac{k_{J2}}{r^5} - \frac{5k_{J2}\sin^2 i \sin^2\theta}{r^5} & \eta_j^2 &= \frac{\mu}{r_j^3} + \frac{k_{J2}}{r_j^5} - \frac{5k_{J2}r_{jZ}^2}{r_j^7} \\ r_j &= \sqrt{(r + x_j)^2 + y_j^2 + z_j^2} & r_{jZ} &= (r + x_j)\sin i \sin\theta + y_j \sin i \cos\theta + z_j \cos i \end{aligned} \tag{41}$$

$$\begin{aligned} \omega_x &= -\frac{k_{J2}\sin 2i \sin\theta}{hr^3} & \omega_y &= 0 & \omega_z &= \frac{h}{r^2} \\ \alpha_x &= -\frac{k_{J2}\sin 2i \cos\theta}{r^5} + \frac{3v_x k_{J2}\sin 2i \sin\theta}{r^4 h} - \frac{8k_{J2}^2 \sin^3 i \cos i \sin^2\theta \cos\theta}{r^6 h^2} \\ \alpha_z &= -\frac{2hv_x}{r^3} - \frac{k_{J2}\sin^2 i \sin 2\theta}{r^5} \end{aligned}$$

The orbital parameters of the chief orbit (origin of the LVLH frame) are governed by the following equation of motion with J_2 effects

$$\begin{aligned} \dot{r} &= v_x & \dot{v}_x &= -\frac{\mu}{r^2} + \frac{h^2}{r^3} - \frac{k_{J2}}{r^4}(1 - 3\sin^2 i \sin^2\theta) \\ \dot{h} &= -\frac{k_{J2}\sin^2 i \sin 2\theta}{r^3} & \dot{\Omega} &= -\frac{2k_{J2}\cos i \sin^2\theta}{hr^3} \\ \dot{i} &= -\frac{k_{J2}\sin 2i \sin 2\theta}{2hr^3} & \dot{\theta} &= \frac{h}{r^2} + \frac{2k_{J2}\cos^2 i \sin^2\theta}{hr^3} \end{aligned} \tag{42}$$

Equation (39) can be written as follows

$$\dot{\mathbf{x}}_j = \mathbf{f}(\mathbf{x}_j, \mathbf{oe}) + B\mathbf{u}_j \tag{43}$$

where $B = [\mathbf{0}_{3\times3}\ \mathbf{I}_{3\times3}]^T$. Linearizing $\mathbf{f}(\mathbf{x}_j, \mathbf{oe})$ yields

$$\mathbf{f}(\mathbf{x}_j, \mathbf{oe}) \approx \mathbf{f}(\bar{\mathbf{x}}_j, \mathbf{oe}) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}_j}\right|_{\bar{\mathbf{x}}_j} (\mathbf{x}_j - \bar{\mathbf{x}}_j) = A(\bar{\mathbf{x}}_j, \mathbf{oe})\mathbf{x}_j + c(\bar{\mathbf{x}}_j, \mathbf{oe}) \tag{44}$$

with

$$A(\bar{\mathbf{x}}_j, \mathbf{oe}) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_3 \\ -\left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}_j}\right|_{\bar{\mathbf{x}}_j} & -2\mathbf{S}(\boldsymbol{\omega}) \end{bmatrix}, \quad c(\bar{\mathbf{x}}_j, \mathbf{oe}) = \begin{bmatrix} \mathbf{0}_{3\times1} \\ -\mathbf{g}(\bar{\boldsymbol{\ell}}_j, \mathbf{oe}) + \left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}_j}\right|_{\bar{\mathbf{x}}_j} \bar{\mathbf{x}}_j \end{bmatrix} \tag{45}$$

The discretized state space matrices are defined as

$$A_d(\bar{\mathbf{x}}_j, \mathbf{oe}) = e^{A(\bar{\mathbf{x}}_j, \mathbf{oe})\Delta t}, \qquad B_d = \int_0^{\Delta t} e^{A(\bar{\mathbf{x}}_j, \mathbf{oe})\tau} B\, d\tau, \qquad c_d = c(\bar{\mathbf{x}}_j, \mathbf{oe})\Delta t \tag{46}$$

REFERENCES

[1] S.-J. Chung and F. Y. Hadaegh, "Swarms of Femtosats for Synthetic Aperture Applications," *Proceedings of the Fourth International Conference on Spacecraft Formation Flying Missions & Technologies*, St-Hubert, Quebec, May 2011.

[2] D. Morgan, S.-J. Chung, L. Blackmore, B. Acikmese, D. Bayard, and F. Y. Hadaegh, "Swarm-Keeping Strategies for Spacecraft Under J_2 and Atmospheric Drag Perturbations," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 5, 2012, pp. 1492–1506.

[3] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen, "A Survey of Spacecraft Formation Flying Guidance and Control (Part I): Guidance," *Proceedings of the American Control Conference*, Jun. 2003, pp. 1733–1739.

[4] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen, "A Survey of Spacecraft Formation Flying Guidance and Control (Part II): Control," *Proceedings of the American Control Conference*, Jun. 2004, pp. 2976–2984.

[5] R. M. Murray, "Recent Research in Cooperative Control of Multivehicle Systems," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 51, 2007.

[6] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules," *IEEE Transactions on Automatic Control*, Vol. 48, No. 6, 2003, pp. 2976–2984.

[7] M. G. Earl and R. D'Andrea, "Iterative MILP Methods for Vehicle-Control Problems," *IEEE Transactions on Robotics*, Vol. 21, No. 6, 2005, pp. 1158–1167.

[8] A. V. Rao, "A Survey of Numerical Methods for Optimal Control," *Advances in the Astronautical Sciences*, Vol. 135, 2010, pp. 497–528.

[9] B. A. Conway, *Spacecraft Trajectory Optimization*. New York, USA: Cambridge Univ. Press, 2010.

[10] I. M. Ross and F. Fahroo, "Legendre Pseudospectral Approximations of Optimal Control Problems," *Lecture Notes in Control and Information Systems*, Vol. 295, 2003.

[11] A. Richards, Y. Kuwata, and J. How, "Experimental Demonstration of Real-time MILP Control," *AIAA Guidance, Navigation, and Control Conference*, Austin, Texas, August 2003.

[12] M. P. Vitus, V. Pradeep, G. M. Hoffman, S. L. Waslander, and C. J. Tomlin, "Tunnel-MILP: Path Planning with Sequential Convex Polytopes," *AIAA Guidance, Navigation, and Control Conference*, Honolulu, HI, August 2008.

[13] M. P. Vitus, S. L. Waslander, and C. J. Tomlin, "Locally Optimal Decomposition for Autonomous Obstacle Avoidance with the Tunnel-MILP Algorithm," *IEEE Conference on Decision and Control*, 2008, pp. 540–545.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[15] J. Mattingley, Y. Wang, and S. Boyd, "Receding Horizon Control: Automatic Generation of High-Speed Solvers," *IEEE Control Systems Magazine*, Vol. 31, No. 3, 2011, pp. 52–65.

[16] B. Acikmese, D. P. Scharf, E. A. Murray, and F. Y. Hadaegh, "A Convex Guidance Algorithm for Formation Reconfiguration," *AIAA Guidance, Navigation, and Control Conference*, Keystone, Colorado, August 2006.

[17] D. Morgan, S.-J. Chung, and F. Y. Hadaegh, "Spacecraft Swarm Guidance Using a Sequence of Decentralized Convex Optimizations," *AIAA/AAS Astrodynamics Specialist Conference*, Minneapolis, MN, August 2012. AIAA 2012-4583.

[18] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A Trust Region Method Based on Interior Point Techniques for Nonlinear Programming," *Math. Program. A*, Vol. 89, No. 1, 2000, pp. 149–185.

[19] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming (version 1.22)," May 2012. http://cvxr.com/cvx/.

[20] A. Bemporad and M. Morari, "Robust Model Predictive Control: A Survey," *Robustness in Identification and Control*, 1999, pp. 207–226.

[21] M. D. Q., J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, Vol. 36, 2000, pp. 789–814.

[22] A. Richards and J. P. How, "Robust variable horizon model predictive control for vehicle maneuvering," *International Journal of Robust and Nonlinear Control*, Vol. 16, 2006, pp. 333–351.

[23] B. Acikmese, J. M. Carson, and D. S. Bayard, "A robust model predictive control algorithm for incrementally conic uncertain/nonlinear systems," *International Journal of Robust and Nonlinear Control*, Vol. 21, 2011, pp. 563–590.

[24] A. Richards and J. How, "Robust Model Predictive Control with Imperfect Information," *American Control Conference*, Portland, OR, June 2005.

[25] Y. Wang and S. Boyd, "Fast Model Predictive Control Using Online Optimization," *IEEE Transactions on Control Systems Technology*, Vol. 18, No. 2, 2010, pp. 267–278.

[26] M. Canale, V. Cerone, D. Piga, and D. Regruto, "Fast implementation of model predictive control with guaranteed performance," *IEEE Conference on Decision and Control*, Orlando, FL, December 2011.

[27] J. M. Carson and B. Acikmese, "A Model Predictive Control Technique with Guaranteed Resolvability and Required Thruster Silent Times for Small-Body Proximity Operations," *AIAA Guidance, Navigation, and Control Conference*, Keystone, CO, August 2006.

[28] G. Xu and D. Wang, "Nonlinear Dynamic Equations of Satellite Relative Motion Around an Oblate Earth," *Journal of Guidance, Control, and Dynamics*, Vol. 31, Sep.-Oct. 2008, pp. 1521–1524.