(Preprint) AAS 22-795

NONLINEAR SPACECRAFT FORMATION FLYING USING CONSTRAINED DIFFERENTIAL DYNAMIC PROGRAMMING

Tomohiro Sasaki*, Koki Ho†, and E. Glenn Lightsey ‡

The advancement of spacecraft guidance, navigation, and control (GNC) technology is essential for future space systems. This paper contributes to the GNC area by solving nonlinear unconstrained/constrained multi-spacecraft optimal control problems using an existing technique of dynamic programming, differential dynamic programming (DDP). DDP is a trajectory optimization methodology that iteratively finds a local optimal control policy around nominal state and control sequences. This method is extensively getting attention in Robotics and Aerospace and is extended to a constrained problem in a recent decade. The constrained DDP (CDDP) has proven its optimality and displayed satisfactory numerical performance. This paper utilizes this algorithm and simulated spacecraft formation flying by the Julia Language. Benefiting from the fast computing language, successful CDDP spacecraft formation flying simulation results are shown in this paper without using any numerical optimization solvers.

INTRODUCTION

Spacecraft formation flying is getting attention as a key technology for advanced space systems. This paper refers to formation flying as a set of more than one spacecraft, each following a common control law.[1,2] In recent decades, several space missions have successfully demonstrated spacecraft formation flying, such as PRISMA (German Aerospace Center),[3] TanDEM-X/TerraSAR-X (German Aerospace Center),[4] GRACE (NASA and German Aerospace Center),[5] ETS-VII (JAXA),[6] etc. These missions have successfully displayed the benefits of formation flying relative to a mission that would be conducted by a single spacecraft. At present, ambitious formation flying missions are under development, such as mDot (NASA),[7] VISORS (multi-university),[8] FFSAT (The University of Tokyo),[9] and LISA (ESA and NASA).[10] Some spacecraft are currently fulfilling their mission.[11,12] Making further progress in formation flying technology with those missions, future missions that are not yet known can be anticipated.

Multi-spacecraft formation flying is one of the challenging areas in the spacecraft formation's guidance, navigation, and control (GNC). As advantages of multi-spacecraft formation flying over monolithic spacecraft, it could achieve a more complex mission than that with a single spacecraft. Furthermore, multi-spacecraft formation flying might be able to compensate for the failure of one or some spacecraft. However, there are disadvantages to distributed spacecraft systems, such as increased collision risk and computational cost/complexity. Additionally, designing a strategy for controlling more than one spacecraft at once is a highly complex problem, but it needs to be solved

*Ph.D. Student, Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA.
†Associate Professor, Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA.
‡David Lewis Professor of Space Systems Technology, Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA.

for future ambitious missions. Significant applications of multi-spacecraft formation flying are, for example, formation de-orbiting, synthesized aperture for the Earth or exoplanets, and distributed sensing.[8–10]

In previous literature, many GNC approaches have been developed for solving nonlinear and linear optimal control problems in the formation flying research area. Here, this paper refers spacecraft relative motion trajectory optimization problem as an optimal control problem. Those approaches are mainly divided into two categories: the indirect method[13–15] and the direct method.[16–18] Direct methods are more popular for solving nonlinear optimal control problems since indirect methods require deriving the first-order necessary conditions for optimality. In contrast, direct methods satisfy their optimality through the direct optimization process.[19] Moreover, some of these trajectory optimization approaches can handle safety constraints and fuel consumption. In addition to these offline trajectory optimization methods, in recent years, model predictive control (MPC) is appealing because of its capability (or performance) for solving spacecraft optimal control problems.[16,18,20–22] MPC computes the control policy by solving a finite-horizon optimization problem subject to state and control constraints regarding current states as initial states. Due to the property of updating control inputs for actuators, MPC can be a stable and robust controller to nonlinear and linear optimal control problems.

This paper is built upon recent progress on one of the trajectory optimization techniques, the differential dynamic programming (DDP) introduced by Jacobson and Mayne.[23] DDP is an offline trajectory optimization methodologies which iteratively finds a local optimal control policy computed from a nominal state and control trajectories. As the name suggests, DDP is an indirect method based on Bellman's principle of optimality, which splits the main optimization problem into smaller optimization subproblems. It has been shown that DDP displays local quadratic convergence under some mild assumptions.[24] Recent works suggest that DDP can be used for min-max type problems,[25,26] model predictive control framework,[27,28] and continuous-time formulation.[26] Although the original DDP requires second-order derivatives of the value function, dynamics, and computational cost, variations of DDP can avoid those difficulties such as the iterative linear quadratic regulator (iLQR) by linear approximations,[29] unscented DDP by the unscented transform,[30] and sampled DDP by the Monte Carlo method.[31]

Furthermore, more studies have been conducted on constrained DDP problems in recent years. Constrained DDP is mainly solved through the two distinct approaches: 1) barrier and augmented Lagrangian method, and 2) active set method. The former directly benefits from the barrier or augmented Lagrangian functions to transform the nonlinear constrained optimization problem into the unconstrained optimization problem.[32,33] The latter benefits from the active set method to take into account for nonlinear constraints once active constraints are identified at each time step.[34] The hybrid method of those algorithms is already reported.[35] Finally, the primal-dual interior-point DDP (IPDDP) is developed,[36] which utilizes a primal-dual interior-point method and proved quadratic convergence properties. This paper utilizes IPDDP under the achievement of Pavlov et.al.'s work.

This paper is organized as follows: The dynamical system of relative motion is described in the first section. The backgrounds of DDP and CDDP are presented in the second section. Finally, numerical simulation results of spacecraft formation flying using differential dynamic programming are shown in the last section.

BACKGROUNDS

Dynamical System

As shown in Figure 1, two different coordinate frames need to be defined to appropriately solve an optimal control problem for spacecraft relative motion around the Earth. At first, the Earth-centered inertial frame (ECI) coordinate frame is defined to locate the position of the chief spacecraft. The ECI frame is an inertially fixed coordinate frame, and its center is located at the center of the Earth. The $\hat{i}$ direction points toward the vernal equinox, the $\hat{k}$ direction points toward the North pole, and the $\hat{j}$ direction completes the right-handed coordinate frame. Thus, $\hat{j}$ is perpendicular to the other two directions. Next, Hill frame is defined to locate the position of the deputy spacecraft with respect to the chief spacecraft. Hill frame is a locally fixed coordinate frame, and its center is located at the center of the chief spacecraft. The $\hat{x}$ direction is aligned with the chief position vector pointing away from the Earth, the $\hat{z}$ direction is aligned with the angular momentum vector, and the $\hat{y}$ direction completes the right-handed coordinate frame. Conventionally, $\hat{x},\ \hat{y},\ \hat{z}$ directions are called the radial, along-track, and cross-track directions. The angular velocity of Hill frame is defined as follows:

$$\boldsymbol{\omega} = \omega_x \hat{\boldsymbol{x}} + \omega_z \hat{\boldsymbol{z}} \tag{1}$$

where ω_x is the rotation rate about the radial direction, and ω_z is the rotation rate about the cross-track direction.

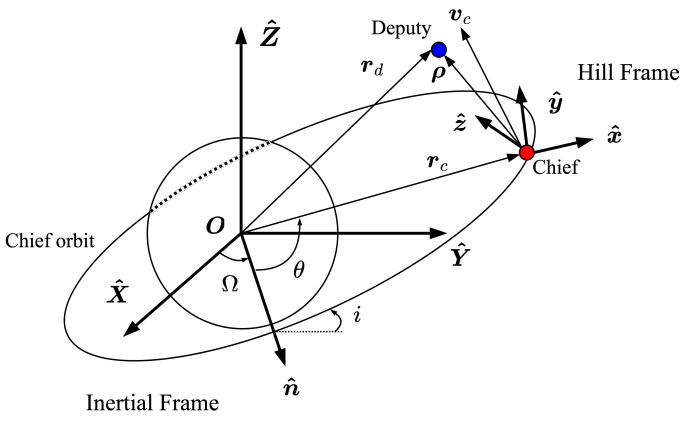

Figure 1. ECI and LVLH frames

In this paper, the multi-agent nonlinear optimal control problem in the chief and multi-deputy systems is defined as well. Thus, the j-th deputy spacecraft state in the LVLH frame is given by $\boldsymbol{x}_j = [x_j,\ y_j,\ z_j,\ \dot{x}_j,\ \dot{y}_j,\ \dot{z}_j]^{\mathrm{T}}$. The nonlinear equation of relative motion for the j-th deputy spacecraft is defined as:[37]

$$\dot{\boldsymbol{x}}_j = \boldsymbol{f}(\text{œ}, \boldsymbol{x}_j, \boldsymbol{u}_j) \tag{2}$$

where $\text{œ} = [r, v_x, h_{\mathrm{ang}}, i, \Omega, \theta]^{\mathrm{T}}$ are the orbital elements of the chief orbit, r is the magnitude of the position vector, v_x is the magnitude of the radial velocity vector, h_{ang} is the magnitude of the

specific angular momentum, i is the inclination, Ω is the right ascension of the ascending node, and θ is the argument of latitude. The equation of motion for the chief spacecraft is defined as

$$\dot{\mathbf{oe}} = \boldsymbol{f}_{\text{chief}}(\mathbf{oe}, \boldsymbol{u}_{\text{chief}}) \tag{3}$$

Note that Equation (4) is a function of its dynamics, the dynamics of the chief spacecraft, and the control input. The following optimal control problem is constructed upon two assumptions. The first assumption is that we only design deputy control inputs, i.e. $\boldsymbol{u}_{\text{chief}} = 0$. The second assumption is that the chief orbital elements and deputy states are known without error.

DIFFERENTIAL DYNAMIC PROGRAMMING

This section introduces the concept and derivation of differential dynamic programming (DDP) and the concept and derivation of interior-point constrained differential dynamic programming (IP-CDDP).

Unconstrained Differential Dynamic Programming

Firstly, an optimal control problem is defined. A nonlinear dynamical system that is described by the following discrete-time differential equation is considered .

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{4}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^m$ denote the state and control input at time step k, respectively. $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ denotes the nonlinear state transition function. We assume that this state transition function $\boldsymbol{f}$ is twice differentiable. A trajectory $\{\boldsymbol{X}, \boldsymbol{U}\}$ is a sequence of states $\boldsymbol{X} = [\boldsymbol{x}_0, \boldsymbol{x}_1, \dots \boldsymbol{x}_N]$ and corresponding control inputs $\boldsymbol{U} = [\boldsymbol{u}_0, \boldsymbol{u}_1, \dots \boldsymbol{u}_{N-1}]$ satisfying Equation (4). The finite-horizon total cost over the horizon N is given by the sum of running cost $\ell(\boldsymbol{x}, \boldsymbol{u}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ and terminal cost $\phi(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ as

$$J\left(\boldsymbol{x}_0, \boldsymbol{U}\right) = \phi\left(\boldsymbol{x}_N\right) + \sum_{k=0}^{N-1} \ell\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) \tag{5}$$

where it is assumed that $\ell(\boldsymbol{x}, \boldsymbol{u})$ and $\phi(\boldsymbol{x})$ are twice differentiable. It is aimed to find an optimal control trajectory that minimizes the cost function under an unconstrained optimal control problem

$$\begin{aligned} \min_{\boldsymbol{U}} J(\boldsymbol{X}, \boldsymbol{U}) = \min_{\boldsymbol{U}} & \left[\phi\left(\boldsymbol{x}_N\right) + \sum_{k=0}^{N-1} \ell\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right)\right] \\ \text{subject to} \quad & \boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \end{aligned} \tag{6}$$

Here the value function is define , which is the minimum cost-to-go at each state and time:

$$V_k(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k} \left[\phi\left(\boldsymbol{x}_N\right) + \sum_{k=0}^{N-1} \ell\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right)\right] \tag{7}$$

Using Bellman's principle of optimality, the sequence of optimal control problems can be expressed as

$$V_k(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k} \left[\ell(\boldsymbol{x}_k, \boldsymbol{u}_k) + V_{k+1}(\boldsymbol{x}_{k+1})\right] \tag{8}$$

with the boundary condition

$$V_N = \phi(\boldsymbol{x}_N) \tag{9}$$

Now the nonlinear optimal control problem is defined.

Next, the algorithm of DDP is introduced. DDP iteratively finds a local optimal control sequence by solving the nonlinear optimal control problem defined above. At each iteration step, DDP executes two steps: 1) backward pass; 2) forward pass, about the nominal state and control trajectories $\{\boldsymbol{X}, \boldsymbol{U}\}$. In the backward pass, the value function is quadratically approximated about given nominal trajectories to find optimal deviations of nominal trajectories. In the forward pass, new state and control trajectories are produced based on the deviations. This set of processes is repeated until given convergence.

Backward pass We define an action-value function $Q(\boldsymbol{x}, \boldsymbol{u}) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ as

$$Q_k(\boldsymbol{x}_k, \boldsymbol{u}_k) = \ell(\boldsymbol{x}_k, \boldsymbol{u}_k) + V_{k+1}(\boldsymbol{x}_{k+1}) \tag{10}$$

This Q function can be quadratically approximated around the nominal trajectory $\{\boldsymbol{X}, \boldsymbol{U}\}$:

$$\begin{aligned} Q(\bar{\boldsymbol{x}} + \delta\boldsymbol{x}, \bar{\boldsymbol{u}} + \delta\boldsymbol{u}) \;\approx\; & Q(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}}) + \begin{bmatrix} Q_{\boldsymbol{x}} \\ Q_{\boldsymbol{u}} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\boldsymbol{u} \end{bmatrix} \\ & + \frac{1}{2} \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\boldsymbol{u} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Q_{\boldsymbol{xx}} & Q_{\boldsymbol{xu}} \\ Q_{\boldsymbol{xu}}^{\mathrm{T}} & Q_{\boldsymbol{uu}} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\boldsymbol{u} \end{bmatrix} \end{aligned} \tag{11}$$

where $\delta\boldsymbol{x}$ and $\delta\boldsymbol{u}$ are the deviations about the nominal trajectories. For the notation simplicity, the time step indices are neglected from now on, and the subscripts indicate the gradient operator, i.e. $\nabla_{\boldsymbol{x}} Q = Q_{\boldsymbol{x}}$. The derivatives of the action-value function Q are given by

$$Q_{\boldsymbol{x}} = \ell_{\boldsymbol{x}} + \boldsymbol{f}_{\boldsymbol{x}}^{\mathrm{T}} V'_{\boldsymbol{x}} \tag{12}$$

$$Q_{\boldsymbol{u}} = \ell_{\boldsymbol{u}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}} V'_{\boldsymbol{x}} \tag{13}$$

$$Q_{\boldsymbol{xx}} = \ell_{\boldsymbol{xx}} + \boldsymbol{f}_{\boldsymbol{x}}^{\mathrm{T}} V'_{\boldsymbol{xx}} \boldsymbol{f}_{\boldsymbol{x}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{xx}} \tag{14}$$

$$Q_{\boldsymbol{xu}} = \ell_{\boldsymbol{xu}} + \boldsymbol{f}_{\boldsymbol{x}}^{\mathrm{T}} V'_{\boldsymbol{xx}} \boldsymbol{f}_{\boldsymbol{u}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{xu}} \tag{15}$$

$$Q_{\boldsymbol{uu}} = \ell_{\boldsymbol{uu}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}} V'_{\boldsymbol{xx}} \boldsymbol{f}_{\boldsymbol{u}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{uu}} \tag{16}$$

where the prime symbol indicates the next time step, i.e. $V'_{\boldsymbol{x}} = V_{k+1}(\boldsymbol{x}_{k+1})$ and "$\cdot$" denotes the tensor dot product. The last terms of $Q_{\boldsymbol{xx}}, Q_{\boldsymbol{xu}}, Q_{\boldsymbol{uu}}$ capture nonlinear changes of the dynamics. Thus, those terms can be neglected when the algorithm pursues faster convergence by disregarding nonlinear solution fidelity. In this case, DDP is called iterative Linear-Quadratic-Regulator (iLQR). iLQR has an advantage on numerical performance since calculations of the nonlinear terms are tensor operations, which are highly computationally expensive with complex dynamics and high-dimensional states. Once the derivatives of the action-value function are stored, the action-value function can be explicitly minimized with respect to the control deviation $\delta\boldsymbol{u}$ given by

$$\delta\boldsymbol{u}^* = \arg\min_{\delta\boldsymbol{u}} Q(\boldsymbol{x} + \delta\boldsymbol{x},\ \boldsymbol{u} + \delta\boldsymbol{u}) = \boldsymbol{k} + \boldsymbol{K}\delta\boldsymbol{x} \tag{17}$$

$$\text{with} \quad \boldsymbol{k} = -Q_{\boldsymbol{uu}}^{-1} Q_{\boldsymbol{u}},\ \boldsymbol{K} = -Q_{\boldsymbol{uu}}^{-1} Q_{\boldsymbol{ux}} \tag{18}$$

Here, $\boldsymbol{k}$ is the feed-forward gain matrix and K is the feedback gain matrix. Lastly, the optimal control deviation can be plugged into Equation (11)

$$V_{\boldsymbol{x}} = Q_{\boldsymbol{x}} + \boldsymbol{K}^{\mathrm{T}} Q_{\boldsymbol{uu}} \boldsymbol{k} + \boldsymbol{K}^{\mathrm{T}} Q_{\boldsymbol{u}} + Q_{\boldsymbol{ux}}^{\mathrm{T}} \boldsymbol{k} \tag{19}$$

$$V_{\boldsymbol{xx}} = Q_{\boldsymbol{xx}} + \boldsymbol{K}^{\mathrm{T}} Q_{\boldsymbol{uu}} \boldsymbol{K} + \boldsymbol{K}^{\mathrm{T}} Q_{\boldsymbol{ux}} + Q_{\boldsymbol{ux}}^{\mathrm{T}} \boldsymbol{K} \tag{20}$$

The backward pass begins from evaluating the value function at the terminal step and proceeds backward in time.

Forward pass After successfully completing the backward pass, the algorithm proceeds to the next step, the forward pass. New trajectories are propagated by updating the nominal state and control trajectories based on information on the optimal control deviation in Equation (18). Thus, we obtain the following update formula:

$$\boldsymbol{x}_0^{\mathrm{new}} = \boldsymbol{x}_0 \tag{21}$$

$$\boldsymbol{u}_k^{\mathrm{new}} = \boldsymbol{u}_k + \gamma\, \boldsymbol{k}_k + \boldsymbol{K}_k (\boldsymbol{x}_k^{\mathrm{new}} - \boldsymbol{x}_k) \tag{22}$$

$$\boldsymbol{x}_{k+1}^{\mathrm{new}} = \boldsymbol{f}(\boldsymbol{x}_k^{\mathrm{new}},\ \boldsymbol{u}_k^{\mathrm{new}}) \tag{23}$$

where γ is a backtracking search parameter and set to be an appropriately small number, and then it is iteratively reduced. Finally, this backward-forward process is repeated until the locally optimal trajectory is converged.

Constrained Differential Dynamic Programming

The CDDP algorithm extends the standard unconstrained DDP to take into account the effect of inequality constraints. This paper follows the interior-point CDDP introduced by Pavlov et. al (2021). The constraints in CDDP should be twice continuously differentiable and are a function of state or/and control. Then the original unconstrained optimal control problem is transformed into:

$$\begin{aligned} \min_{\boldsymbol{U}} J(\boldsymbol{X}, \boldsymbol{U}) = \min_{\boldsymbol{U}} & \left[\phi\left(\boldsymbol{x}_N\right) + \sum_{k=0}^{N-1} \ell\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) \right] \\ \text{subject to} \quad & \boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ & \boldsymbol{c}(\boldsymbol{x}_k, \boldsymbol{u}_k) \leq 0 \end{aligned} \tag{24}$$

where $\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{u}) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^d$ is twice continuously differentiable constraint function. Using Bellman's principle of optimality, the sequence of optimal control problems can be expressed as

$$V_k\left(\boldsymbol{x}_k\right) = \min_{\boldsymbol{u}_k \text{ s.t. } \boldsymbol{c}(\boldsymbol{x}, \boldsymbol{u}) \leq 0} \left[\ell\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) + V_{k+1}\left(\boldsymbol{x}_{k+1}\right) \right] \tag{25}$$

with the boundary condition

$$V_N = \phi(\boldsymbol{x}_N) \tag{26}$$

Here, the existing minimax DDP technique[25,36] is introduced to transform the original sequential minimization problems into sequential minimax problems as follows:

$$V_k\left(\boldsymbol{x}_k\right) = \min_{\boldsymbol{u}_k} \max_{\boldsymbol{\lambda}_k \geq 0} \left[\mathcal{L}\left(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_k\right) + V_{k+1}\left(\boldsymbol{x}_{k+1}\right) \right] \tag{27}$$

where $\mathcal{L}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_k) : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}$ is the Lagrangian function defined as

$$\mathcal{L}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_k) = \ell(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{\lambda}_k^{\mathrm{T}} \boldsymbol{c}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{28}$$

with the corresponding Lagrange multiplier $\boldsymbol{\lambda} \in \mathbb{R}^d$. As the standard DDP finds an optimal solution, CDDP iteratively finds a local optimal control sequence by solving the constrained nonlinear optimal control problem defined above. At each iteration step, DDP executes two steps: 1) backward pass; 2) forward pass, about the nominal state, control, and Lagrange multiplier trajectories $\{\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{\Lambda}\}$. In the backward pass, the value function is quadratically approximated about given nominal trajectories to find optimal deviations of nominal trajectories. In the forward pass, new state, control, and Lagrange multiplier trajectories are propagated based on the deviations. This set of processes is repeated until given convergence. The feasibility is ensured in the forward pass. Note that CDDP does not always have an initial feasible trajectory, unlike the standard DDP. This paper assumes that the initial feasible trajectory is given, however, the interior-point CDDP can deal with the initial infeasible trajectory. The inquisitive reader can find the detail in proof of optimality regarding the interior point DDP in Reference 36

Backward Pass Define an action-value function $Q(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}$ as

$$Q_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_k) = \mathcal{L}_k(\boldsymbol{x}_k, \boldsymbol{u_k}, \boldsymbol{\lambda}_k) + V_{k+1}(\boldsymbol{x}_{k+1}) \tag{29}$$

The quadratically approximated Q function around the nominal trajectory $\{\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{\Lambda}\}$ is given by

$$\begin{aligned} Q(\boldsymbol{x} + \delta\boldsymbol{x}, \boldsymbol{u} + \delta\boldsymbol{u}, \boldsymbol{\lambda} + \delta\boldsymbol{\lambda}) \;\approx\; & Q(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) + \begin{bmatrix} Q_{\boldsymbol{x}} \\ Q_{\boldsymbol{u}} \\ Q_{\boldsymbol{\lambda}} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\boldsymbol{u} \\ \delta\boldsymbol{\lambda} \end{bmatrix} \\ & + \frac{1}{2} \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\boldsymbol{u} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Q_{\boldsymbol{xx}} & Q_{\boldsymbol{xu}} & Q_{\boldsymbol{x\lambda}} \\ Q_{\boldsymbol{xu}}^{\mathrm{T}} & Q_{\boldsymbol{uu}} & Q_{\boldsymbol{u\lambda}} \\ Q_{\boldsymbol{x\lambda}}^{\mathrm{T}} & Q_{\boldsymbol{u\lambda}}^{\mathrm{T}} & Q_{\boldsymbol{\lambda\lambda}} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{x} \\ \delta\boldsymbol{u} \\ \delta\boldsymbol{\lambda} \end{bmatrix} \end{aligned} \tag{30}$$

The derivatives of the action-value function Q are given by

$$Q_{\boldsymbol{x}} = \mathcal{L}_{\boldsymbol{x}} + \boldsymbol{f}_{\boldsymbol{x}}^{\mathrm{T}} V'_{\boldsymbol{x}} \tag{31}$$
$$Q_{\boldsymbol{u}} = \mathcal{L}_{\boldsymbol{u}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}} V'_{\boldsymbol{x}} \tag{32}$$
$$Q_{\boldsymbol{xx}} = \mathcal{L}_{\boldsymbol{xx}} + \boldsymbol{f}_{\boldsymbol{x}}^{\mathrm{T}} V'_{\boldsymbol{xx}} \boldsymbol{f}_{\boldsymbol{x}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{xx}} \tag{33}$$
$$Q_{\boldsymbol{ux}} = \mathcal{L}_{\boldsymbol{ux}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}} V'_{\boldsymbol{xx}} \boldsymbol{f}_{\boldsymbol{x}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{ux}} \tag{34}$$
$$Q_{\boldsymbol{uu}} = \mathcal{L}_{\boldsymbol{uu}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}} V'_{\boldsymbol{xx}} \boldsymbol{f}_{\boldsymbol{u}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{uu}} \tag{35}$$
$$Q_{\boldsymbol{\lambda}} = \mathcal{L}_{\boldsymbol{\lambda}} = \boldsymbol{c} \tag{36}$$
$$Q_{\boldsymbol{\lambda x}} = \mathcal{L}_{\boldsymbol{\lambda x}} = \boldsymbol{c}_{\boldsymbol{x}} \tag{37}$$
$$Q_{\boldsymbol{\lambda u}} = \mathcal{L}_{\boldsymbol{\lambda u}} = \boldsymbol{c}_{\boldsymbol{u}} \tag{38}$$
$$Q_{\boldsymbol{\lambda\lambda}} = 0 \tag{39}$$

Now the approximated action-value function is plugged into Equation (27). Then, the minimization problem can be explicitly solved with respect to the control deviation $\delta \boldsymbol{u}$ as

$$Q_{\boldsymbol{u}} + Q_{\boldsymbol{ux}}\delta \boldsymbol{x} + Q_{\boldsymbol{uu}}\delta \boldsymbol{u} + Q_{\boldsymbol{\lambda u}}^{\mathrm{T}}\delta \boldsymbol{\lambda} = 0 \tag{40}$$

On the other hand, the inner maximization problem can also be solved by taking the first-order derivative. It results in the following equation

$$(\boldsymbol{\lambda} + \delta \boldsymbol{\lambda}) \odot (Q_{\boldsymbol{\lambda}} + Q_{\boldsymbol{\lambda x}}\delta \boldsymbol{x} + Q_{\boldsymbol{\lambda u}}\delta \boldsymbol{u}) = 0 \tag{41}$$

where $\odot$ is the element-wise Hadamard product. Ignoring the second-order terms and adding the perturbation vector $\boldsymbol{\mu}$, the following parametric system can be obtained:

$$\begin{bmatrix} Q_{uu} & Q_{us} \\ \mathrm{diag}(\boldsymbol{\lambda})Q_{su} & \mathrm{diag}(\boldsymbol{c}) \end{bmatrix} \begin{bmatrix} \delta u \\ \delta \boldsymbol{\lambda} \end{bmatrix} = - \begin{bmatrix} Q_u \\ \mathrm{diag}(\boldsymbol{\lambda})\boldsymbol{c} + \boldsymbol{\mu} \end{bmatrix} - \begin{bmatrix} Q_{ux} \\ \mathrm{diag}(\boldsymbol{\lambda})\mathrm{Q}_{sx} \end{bmatrix} \delta x \tag{42}$$

The solution to Equation (42) is given by

$$\begin{bmatrix} \boldsymbol{k} & \boldsymbol{K} \\ \boldsymbol{h} & \boldsymbol{H} \end{bmatrix} = \begin{bmatrix} Q_{\boldsymbol{uu}} & Q_{\boldsymbol{us}} \\ \mathrm{diag}(\boldsymbol{\lambda})\mathrm{Q}_{\boldsymbol{su}} & \mathrm{diag}(\boldsymbol{c}) \end{bmatrix}^{-1} \begin{bmatrix} Q_{\boldsymbol{u}} & Q_{\boldsymbol{ux}} \\ \mathrm{diag}(\boldsymbol{\lambda})\boldsymbol{c} + \boldsymbol{\mu} & \mathrm{diag}(\boldsymbol{\lambda})\mathrm{Q}_{\boldsymbol{\lambda x}} \end{bmatrix} \tag{43}$$

Here, $\boldsymbol{k}$ is the feed-forward gain matrix for control policy, $\boldsymbol{K}$ is the feedback gain matrix for control policy, $\boldsymbol{h}$ is the feed-forward gain matrix for the Lagrange multiplier, $\boldsymbol{H}$ is the feedback gain matrix for the Lagrange multiplier. Lastly, the derivative of the value function can be written as

$$V_{\boldsymbol{x}} = \hat{Q}_{\boldsymbol{x}} + \boldsymbol{K}^{\mathrm{T}}\hat{Q}_{\boldsymbol{uu}}\boldsymbol{k} + \boldsymbol{K}^{\mathrm{T}}\hat{Q}_{\boldsymbol{u}} + \hat{Q}_{\boldsymbol{ux}}^{\mathrm{T}}\boldsymbol{k} \tag{44}$$

$$V_{\boldsymbol{xx}} = \hat{Q}_{\boldsymbol{xx}} + \boldsymbol{K}^{\mathrm{T}}\hat{Q}_{\boldsymbol{uu}}\boldsymbol{K} + \boldsymbol{K}^{\mathrm{T}}\hat{Q}_{\boldsymbol{ux}} + \hat{Q}_{\boldsymbol{ux}}^{\mathrm{T}}\boldsymbol{K} \tag{45}$$

where

$$\hat{Q}_{\boldsymbol{x}} = Q_{\boldsymbol{x}} - Q_{xs}\mathrm{diag}(\boldsymbol{C})^{-1}(\mathrm{diag}(\boldsymbol{\lambda})\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\mu}) \tag{46}$$

$$\hat{Q}_{\boldsymbol{u}} = Q_{\boldsymbol{u}} - Q_{\boldsymbol{us}}\mathrm{diag}(\boldsymbol{C})^{-1}(\mathrm{diag}(\boldsymbol{\lambda})\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\mu}) \tag{47}$$

$$\hat{Q}_{\boldsymbol{xx}} = Q_{\boldsymbol{xx}} - Q_{\boldsymbol{xs}}\mathrm{diag}(\boldsymbol{C})^{-1}\mathrm{diag}(\boldsymbol{\lambda})\mathrm{Q}_{\boldsymbol{\lambda x}} \tag{48}$$

$$\hat{Q}_{\boldsymbol{xu}} = Q_{\boldsymbol{xu}} - Q_{\boldsymbol{x\lambda}}\mathrm{diag}(\boldsymbol{C})^{-1}\mathrm{diag}(\boldsymbol{\lambda})\mathrm{Q}_{\boldsymbol{\lambda u}} \tag{49}$$

$$\hat{Q}_{\boldsymbol{uu}} = Q_{\boldsymbol{uu}} - Q_{\boldsymbol{u\lambda}}\mathrm{diag}(\boldsymbol{C})^{-1}\mathrm{diag}(\boldsymbol{\lambda})\mathrm{Q}_{\boldsymbol{\lambda u}} \tag{50}$$

The backward pass in CDDP also begins from evaluating the value function at the terminal step and proceeds backward in time.

Forward Pass After successfully completing the backward pass, the algorithm proceeds to the next step, the forward pass. New trajectories are propagated by updating the nominal state, control, and the multiplier trajectories based on information on the parametric solution in Equation (43). Hence, we obtain the following update formula:

$$\boldsymbol{x}_0^{\text{new}} = \boldsymbol{x}_0 \tag{51}$$
$$\boldsymbol{u}_k^{\text{new}} = \boldsymbol{u}_k + \gamma\, \boldsymbol{k}_k + \boldsymbol{K}_k(\boldsymbol{x}_k^{\text{new}} - \boldsymbol{x}_k) \tag{52}$$
$$\boldsymbol{\lambda}_k^{\text{new}} = \boldsymbol{\lambda}_k + \gamma\, \boldsymbol{h}_k + \boldsymbol{H}_k(\boldsymbol{x}_k^{\text{new}} - \boldsymbol{x}_k) \tag{53}$$
$$\boldsymbol{x}_{k+1}^{\text{new}} = \boldsymbol{f}(\boldsymbol{x}_k^{\text{new}},\ \boldsymbol{u}_k^{\text{new}}) \tag{54}$$

where γ is a backtracking search parameter and set to be an appropriately small number, it is then iteratively reduced. Note that unlike the DDP algorithm, the CDDP algorithm should satisfy $\boldsymbol{c}^{\text{new}}(\boldsymbol{x}, \boldsymbol{u}) < 0$ and $\boldsymbol{\lambda}^{\text{new}} > 0$ at each time step. Finally, this backward-forward process is repeated until the locally optimal trajectory is converged.

Regularization

To guarantee numerical convergence, $Q_{\boldsymbol{uu}}$ must be positive definite over the trajectory. In particular, a regularization scheme introduced in Reference 34, 38 is employed in this paper. The scheme is as follows:

$$Q_{\boldsymbol{xx}} = \mathcal{L}_{\boldsymbol{xx}} + \boldsymbol{f}_{\boldsymbol{x}}^{\mathrm{T}}(V'_{\boldsymbol{xx}} + \nu_1 \boldsymbol{I}_n)\boldsymbol{f}_{\boldsymbol{x}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{xx}} \tag{55}$$
$$Q_{\boldsymbol{ux}} = \mathcal{L}_{\boldsymbol{ux}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}}(V'_{\boldsymbol{xx}} + \nu_1 \boldsymbol{I}_n)\boldsymbol{f}_{\boldsymbol{x}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{ux}} \tag{56}$$
$$Q_{\boldsymbol{uu}} = \mathcal{L}_{\boldsymbol{uu}} + \boldsymbol{f}_{\boldsymbol{u}}^{\mathrm{T}}(V'_{\boldsymbol{xx}} + \nu_1 \boldsymbol{I}_n)\boldsymbol{f}_{\boldsymbol{u}} + V'_{\boldsymbol{x}} \cdot \boldsymbol{f}_{\boldsymbol{uu}} + \nu_2 \boldsymbol{I}_m \tag{57}$$

where ν_1 and ν_2 are positive regularization parameters. Those parameters play the role of the Levenberg-Marquardt parameter. The value ν_1 pushes the new trajectory back to the previous trajectory, improving solution trajectories' robustness. On the other hand, the value ν_2 directly keeps $Q_{\boldsymbol{uu}}$ to be positive definite and makes the control input variations gradual changes.

SIMULATION RESULTS

This section presents numerical simulation results of the spacecraft's optimal control problem given the dynamical system. There are two types of simulation cases 1) unconstrained case and 2) constrained case. Each case includes two spacecraft formations and four spacecraft formations. The initial orbital elements for the chief spacecraft is defined as $\text{æ} = [a, e, i, \Omega, \omega, M] = [6871\ \text{km}, 0.001, 98\ \text{deg}, 10\ \text{deg}, 10\ \text{deg}, 10\ \text{deg}]$. The time step and length are set to $\mathrm{d}t = 10\,\text{s}$ and $t_N = 1080$. The DDP and CDDP simulation results are compared with an existing optimal control algorithm, sequential convex programming (SCP) referring literature.[18,39] All algorithm is implemented using the Julia Language. The SCP algorithm utilized the Julia Language's JuMP.jl, a modeling package. The CDDP performance is improved using the Julia Language's ForwardDiff.jl.[40] The computational performance is measured by BenchmarkTools.jl.

For the simulation, the terminal and running cost functions are defined in a quadratic form:

$$\begin{aligned} \phi(\boldsymbol{x}) &= \boldsymbol{x}^{\mathrm{T}} \boldsymbol{R}_1 \boldsymbol{x} \\ \ell(\boldsymbol{u}) &= \boldsymbol{u}^{\mathrm{T}} \boldsymbol{R}_2 \boldsymbol{u} \end{aligned} \tag{58}$$

where $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ are weight matrices given by

$$\begin{aligned} R_1 &= \begin{bmatrix} 10^{-1}\, \boldsymbol{I}_3 & 0_{3\times 3} \\ 0_{3\times 3} & 10^{2}\, \boldsymbol{I}_3 \end{bmatrix} \\ R_2 &= 10^{6}\, \boldsymbol{I}_3 \end{aligned} \tag{59}$$

Chief and Single Deputy System

First, a single chief and single deputy system is tested to learn a property of the algorithm convergence. The initial and final conditions of the deputy spacecraft are shown in Table 1. The terminal condition is applied as the terminal cost.

Table 1. Initial and Final Conditions

Parameter	Initial value	Final value
x m	-94	-38
y m	68	27
z m	34	14
$\dot{x}$ m/s	0.379	0.015
$\dot{y}$ m/s	0.209	0.083
$\dot{z}$ m/s	0.104	0.042

Unconstrained Case First, we consider an unconstrained spacecraft formation flying problem for the purpose of obtaining general performance of the algorithms. Note that inactive control constraints are added to CDDP and SCP for better numerical stability. The DDP algorithm is also implemented as a reference. The results of unconstrained trajectories are shown in Figure 2. All trajectory approaches the final position however, the CDDP (DDP) trajectory is slightly different from the SCP one. The transitions of cost over the iteration are shown in Figure 3. The cost for all three significantly decreases within a few iterations. Table 2 shows the fuel cost for all trajectories. The fuel cost for both the CDDP and DDP trajectories was 0.077 m/s, while the cost for the SCP trajectory was 0.12 m/s. Finally, Table 3 shows the benchmark performance of CDDP, DDP, and SCP algorithms. It results that CDDP's performance time is approximately five times faster than SCP's.

Control Constraints Case Next, we consider a control-limited spacecraft formation flying problem. This case is designed to find the proper nature of constrained optimal control. The control constraint is set to $u_{\max} = 4 \times 10^{-6}$ m/s^2 and $u_{\min} = -4 \times 10^{-6}$ m/s^2. When the control constraints are applied, the CDDP trajectory differs from the SCP trajectory as shown in Figure . The CDDP trajectory rendezvous with the chief spacecraft and then approaches the final position. Figure 5 shows the control inputs history. The CDDP and SCP control sequences are similar to each other. Both control sequences are applied with a burn-burn scheme. The fuel cost for the CDDP and DDP trajectories was 0.074 m/s, whereas the cost for the SCP trajectory was 0.075 m/s as shown in Table 4. The benchmark performance of the CDDP algorithm was 14.8 s, whereas the benchmark

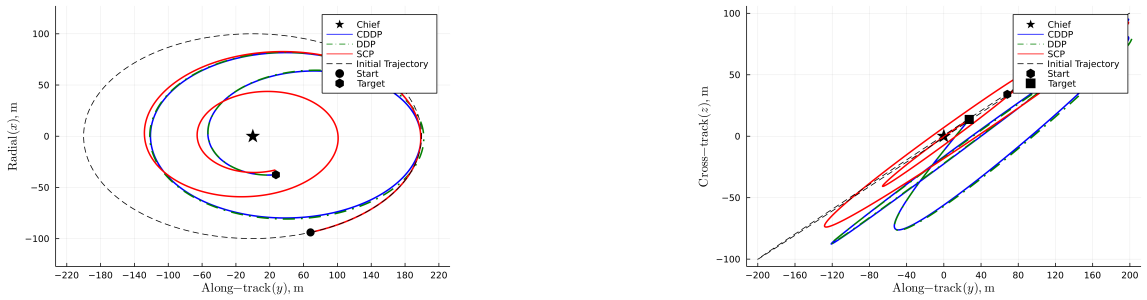

Figure 2. Unconstrained Trajectories: (left) Along-track and Radial plane; (right) Along-track and Cross-track plane

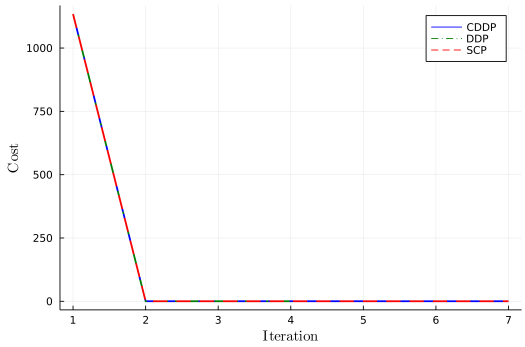

Figure 3. Cost Transitions

Table 2. Total ΔV Cost

Method	Total ΔV (m/s)
CDDP	0.077
SCP	0.12
DDP (Ref)	0.077

Table 3. Benchmark Performance

Method	Benchmark Time (s)
CDDP	8.68
SCP	41.3
DDP (Ref)	3.63

performance of the SCP algorithm was 140 s. The performance difference between CDDP and SCP becomes larger when the constraints are applied.

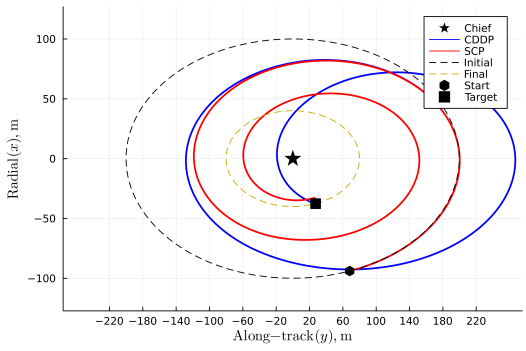

Figure 4. Control Limited Trajectories

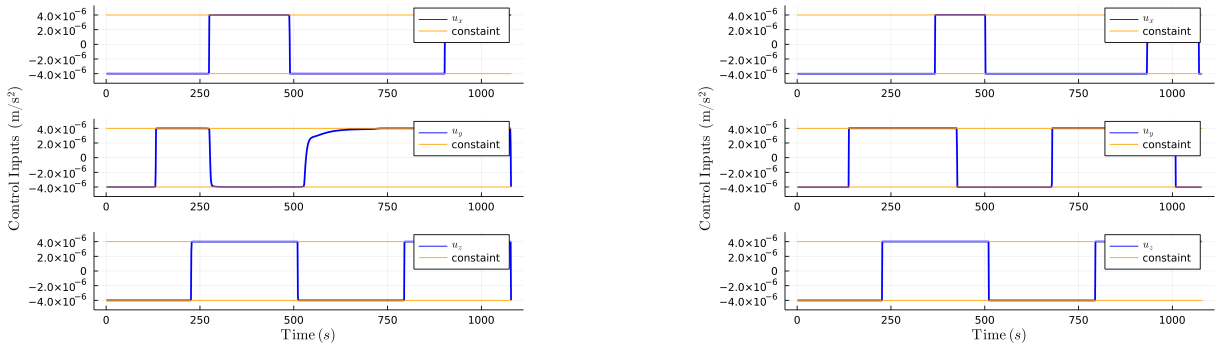

Figure 5. Control Inputs History: (left) CDDP; (right) SCP

Table 4. Total ΔV Cost

Method	Total ΔV (m/s)
CDDP	0.074
SCP	0.075

Table 5. Benchmark Performance

Method	Benchmark Time (s)
CDDP	14.8
SCP	140

Control and State Constraints Next, we consider a more complex problem. The control constraints and a state safe constraint are designed. Due to obvious performance differences, the SCP trajectory is not propagated. The safe state constraint is defined as:

$$39 - \sqrt{x^2 + y^2} \leq 0 \tag{60}$$

It implies that the constrained trajectory never intrudes into a circle with a radius of 39 m around the chief spacecraft. The resulting trajectory is shown in Figure 6. The CDDP trajectory was pushed off apart from the chief spacecraft and lay on the edge of the safe constraint. Due to the control limit and severe safe constraint, the terminal position error results in a larger value than the control limited

case. Figure 7 shows the control input history. Since the trajectory does not strictly reach the final position, it saves control inputs. The fuel cost for this trajectory was 0.052 m/s. The benchmark performance was 46.1 s. Compared to the control-limited case, the benchmark gets worse. This is because algorithm descending was very slow when the algorithm computes the trajectory around the safe constraint.

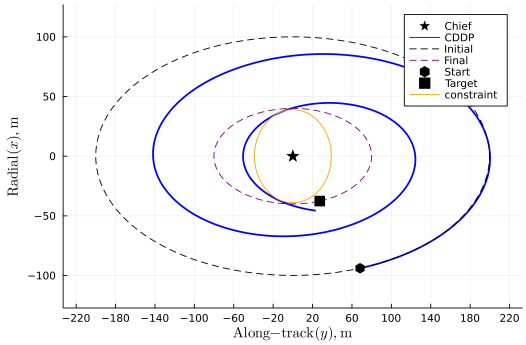

Figure 6. Control and State Constrained trajectories

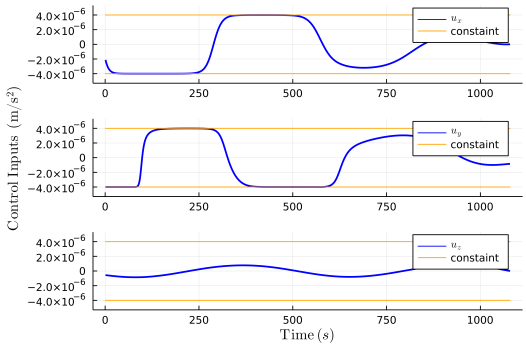

Figure 7. Control and State Constrained Control Input History

Single-Chief and Triple-Deputy System

Finally, we consider a multi-spacecraft constrained (control and state) problem. The initial and final conditions of deputy spacecraft are shown in Table 6 The control constraint is set to $u_{\max} = 4 \times 10^{-6}$ m/s^2 and $u_{\min} = -4 \times 10^{-6}$ m/s^2. The safe constraint distance is set to 25 m. Collision avoidance constraints between each deputy are not considered since they might break the feasible initial solution assumption. Figure 8 shows the CDDP trajectories of deputies under control and state constraints. All trajectory approaches their final state. The deputy # 2 trajectory lies on the state constraint, while the other two are free from the state constraint. Since the safe constraints between deputies are not applied, the deputies are very close to each other at some time steps. Figure 9 shows the multi-spacecraft control input history.

Table 6. Initial and Final Conditions

Deputy #	#1		#2		#3	
Parameter	Initial value	Final value	Initial value	Final value	Initial value	Final value
x m	-94	-38	85	28	-43	-26
y m	68	27	-61	-20	-180	-87
z m	34	14	85	-10	-99	-43
$\dot{x}$ m/s	0.38	0.015	-0.034	-0.011	-0.10	-0.048
$\dot{y}$ m/s	0.21	0.083	-0.19	-0.063	0.097	-0.059
$\dot{z}$ m/s	0.10	0.042	-0.034	0.031	0.052	-0.029

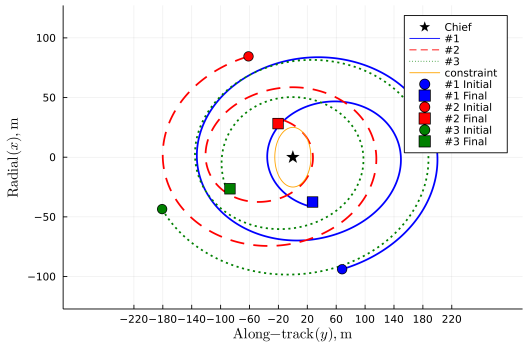

Figure 8. Multi-spacecraft Control and State Constrained trajectories

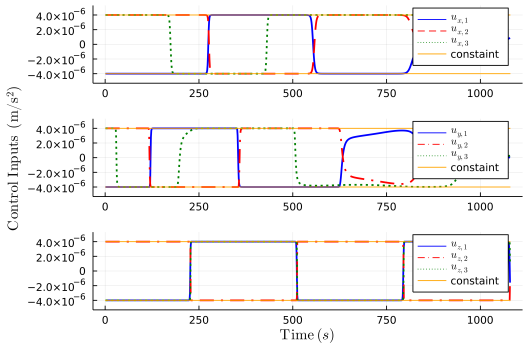

Figure 9. Multi-spacecraft Control and State Constrained Control Input History

CONCLUSION

The multi-spacecraft formation flying problem with control and state constraints is solved using a CDDP algorithm. The problem is set up to satisfy a twice differentiable condition on the dynamics model, terminal function, and cost function. The CDDP algorithm iteratively finds a local optimal control policy around nominal state, control, Lagrange multiplier sequences. At each iteration step CDDP executes two step: backward pass and forward pass. In order to include the effect of constraints in the minimization problem at the backward pass, the minimization problem is transformed into a min-max problem. Then, local optimal deviations of control and Lagrange multiplier can be obtained. After completing the backward pass, new trajectories are propagated in the forward pass. The set of processes is repeated until convergence.

The simulation results of spacecraft formation flying showed successful convergence of optimal control and constraint satisfaction. As a result, the CDDP benchmark performance was faster than the benchmark performance of the SCP algorithm. The simulation results of multi-spacecraft formation flying also showed successful convergence of optimal control and constraint satisfaction.

A future research direction is applying the CDDP algorithm in a predictive control setting to gain the fast and robust trajectory optimization property. To avoid the collision of deputy spacecraft, state constraints between deputies should be considered in future research. A constraint margin should also be considered for a safe spacecraft formation flying mission.

ACKNOWLEDGMENT

The first author has received support from the Nakajima Foundation.

APPENDIX: EQUATIONS OF MOTION OF THE CHIEF AND DEPUTY

We define a constant for the dynamics model as

$$k_{J_2} = \frac{3J_2\mu R_e^2}{2} \tag{61}$$

where μ is the Earth's gravitational constant, J_2 is the second zonal harmonic coefficient of the Earth, and R_e is the Earth's equatorial radius.

The chief equations of motion can be described as follows:[37]

$$\dot{r} = v_x \tag{62}$$

$$\dot{v}_x = -\frac{\mu}{r^2} + \frac{h^2}{r^3} - \frac{k_{J_2}(1 - 3\sin^2 i \sin^2\theta)}{r^4} \tag{63}$$

$$\dot{h} = -\frac{k_{J2}\sin^2 i \sin 2\theta}{r^3} \tag{64}$$

$$\dot{\Omega} = -\frac{k_{J_2}\sin 2i \sin 2\theta}{2hr^3} \tag{65}$$

$$\dot{i} = -\frac{2k_{J_2}\cos i \sin^2\theta}{hr^3} \tag{66}$$

$$\dot{\theta} = \frac{h}{r^2} + \frac{2k_{J_2}\cos^2 i \sin^2\theta}{hr^3} \tag{67}$$

The deputy equations of motion can be described as follows:

$$\ddot{x}_j = 2\dot{y}_j\omega_z - (\eta_j^2 - \omega_z^2)x_j + \alpha_z y - z\omega_x\omega_z - (\zeta_j - \zeta)\sin i \sin\theta - r(\eta_j^2 - \eta^2) - u_{x,\text{chief}} + u_{x_j} \tag{68}$$

$$\ddot{y}_j = -2\dot{x}_j\omega_z + 2\dot{z}_j\omega_x - x_j\alpha_z - y(\eta_j^2 - \omega_z^2 - \omega_x^2) + z_j\alpha_x - (\zeta_j - \zeta)\sin i\cos\theta - u_{y,\text{chief}} + u_{y_j} \tag{69}$$

$$\ddot{z}_j = -2\dot{y}_j\omega_x - x_j\omega_x\omega_z - y_j\alpha_x - z_j(\eta_j^2 - \omega_x^2) - (\zeta_j - \zeta)\cos i - u_{z,\text{chief}} + u_{z_j} \tag{70}$$

where

$$\zeta = \frac{2k_{J_2}\sin i\sin\theta}{r^4} \tag{71}$$

$$\zeta_j = \frac{2k_{J_2}r_{jZ}}{r_j^5} \tag{72}$$

$$\eta^2 = \frac{\mu}{r^3} + \frac{k_{J_2}}{r^5} - \frac{5k_{J_2}\sin^2 i\sin^2\theta}{r^5} \tag{73}$$

$$\eta_j^2 = \frac{\mu}{r_j^3} + \frac{k_{J_2}}{r_j^5} - \frac{5k_{J_2}\sin^2 i\sin^2\theta r_{jZ}^2}{r_j^7} \tag{74}$$

$$r_j = \sqrt{(r + x_j)^2 + y_j^2 + z_j^2} \tag{75}$$

$$r_{rZ} = (r + x_j)\sin i\sin\theta + y_j\sin i\cos\theta + z_j\cos i \tag{76}$$

$$\omega_x = -\frac{k_{J_2}\sin 2i\theta}{hr^3} + \frac{r}{h}u_{z,\text{chief}} \tag{77}$$

$$\omega_z = \frac{h}{r^2} \tag{78}$$

$$\alpha_x = \dot{\omega}_x = -\frac{k_{J_2}\sin 2i\cos\theta}{r^5} + \frac{3v_x k_{J_2}\sin 2i\sin\theta}{r^4 h} - \frac{8k_{J_2}^2\sin^3 i\cos i\sin^2\theta\cos\theta}{r^6 h^2} \tag{79}$$

$$\alpha_z = \dot{\omega}_z = -\frac{2hv_x}{r^3} - \frac{k_{J_2}\sin^2 i\sin 2\theta}{r^5} \tag{80}$$

REFERENCES

[1] 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 2003 American Control Conference*, Vol. 2, Denver, CO, 2003, pp. 1733–1739, 10.1109/acc.2003.1239845.

[2] 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 2004 American Control Conference*, Vol. 4, Boston, MA, 2004, pp. 2976–2985 vol.4, 10.23919/acc.2004.1384365.

[3] E. Gill, O. Montenbruck, and S. D'Amico, "Autonomous Formation Flying for the PRISMA Mission," *Journal of Spacecraft and Rockets*, Vol. 44, No. 3, 2007, pp. 671–681, 10.2514/1.23015.

[4] O. Montenbruck, R. Kahle, S. D'Amico, and J.-S. Ardaens, "Navigation and control of the TanDEM-X formation," *The Journal of the Astronautical Sciences*, Vol. 56, No. 3, 2008, pp. 341–357, 10.1007/bf03256557.

[5] B. D. Tapley, B. E. Schutz, and G. H. Born, *Statistical Orbit Determination*. Elsevier, 2004, 10.1016/b978-0-12-683630-1.x5019-x.

[6] M. Mokuno, I. Kawano, and T. Suzuki, "In-orbit demonstration of rendezvous laser radar for unmanned autonomous rendezvous docking," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 40, No. 2, 2004, pp. 617–626, 10.1109/taes.2004.1310009.

[7] S. D'Amico, A. Koenig, B. Macintosh, and D. Mauro, "System Design of the Miniaturized Distributed Occulter Telescope (mDOT) Science Mission," *Proceedings of the 33rd AIAA/USU Small Satellite Conference*, SSC19-IV-08, Logan, UT, AIAA/USU, 2019.

[8] A. Koenig, S. D'Amico, and E. G. Lightsey, "Formation Flying Orbit and Control Concept for the VISORS Mission," *AIAA Scitech 2021 Forum*, 2021, 10.2514/6.2021-0423.

[9] R. SUZUMOTO, S. IKARI, N. MIYAMURA, and S. NAKASUKA, "μm-class Control of Relative Position and Attitude for a Formation Flying Synthetic Aperture Telescope with Micro-satellites," *TRANSACTIONS OF THE JAPAN SOCIETY FOR AERONAUTICAL AND SPACE SCIENCES*, Vol. 64, No. 2, 2021, pp. 101–111, 10.2322/tjsass.64.101.

[10] P. Amaro-Seoane, H. Audley, S. Babak, J. Baker, E. Barausse, P. Bender, E. Berti, P. Binetruy, M. Born, D. Bortoluzzi, J. Camp, C. Caprini, V. Cardoso, M. Colpi, J. Conklin, N. Cornish, C. Cutler, K. Danzmann, R. Dolesi, L. Ferraioli, V. Ferroni, E. Fitzsimons, J. Gair, L. G. Bote, D. Giardini, F. Gibert, C. Grimani, H. Halloin, G. Heinzel, T. Hertog, M. Hewitson, K. Holley-Bockelmann, D. Hollington, M. Hueller, H. Inchauspe, P. Jetzer, N. Karnesis, C. Killow, A. Klein, B. Klipstein, N. Korsakova, S. L. Larson, J. Livas, I. Lloro, N. Man, D. Mance, J. Martino, I. Mateos, K. McKenzie, S. T. McWilliams, C. Miller, G. Mueller, G. Nardini, G. Nelemans, M. Nofrarias, A. Petiteau, P. Pivato, E. Plagnol, E. Porter, J. Reiche, D. Robertson, N. Robertson, E. Rossi, G. Russano, B. Schutz, A. Sesana, D. Shoemaker, J. Slutsky, C. F. Sopuerta, T. Sumner, N. Tamanini, I. Thorpe, M. Troebs, M. Vallisneri, A. Vecchio, D. Vetrugno, S. Vitale, M. Volonteri, G. Wanner, H. Ward, P. Wass, W. Weber, J. Ziemer, and P. Zweifel, "Laser Interferometer Space Antenna," tech. rep., 2017.

[11] C. W. Roscoe, J. J. Westphal, and E. Mosleh, "Overview and GNC design of the CubeSat Proximity Operations Demonstration (CPOD) mission," *Acta Astronautica*, Vol. 153, 2018, pp. 410–421, 10.1016/j.actaastro.2018.03.033.

[12] G.-N. Kim, S.-Y. Park, T. Lee, D.-E. Kang, S. Jeon, J. Son, N. Kim, Y.-K. Park, and Y. Song, "Development of CubeSat systems in formation flying for the solar science demonstration: The CANYVAL-C mission," *Advances in Space Research*, Vol. 68, No. 11, 2021, pp. 4434–4455, 10.1016/j.asr.2021.09.021.

[13] A. Zavoli and G. Colasurdo, "Indirect Optimization of Finite-Thrust Cooperative Rendezvous," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 2, 2014, pp. 304–314, 10.2514/1.g000531.

[14] A. Heydari, "Optimal Impulsive Control Using Adaptive Dynamic Programming and Its Application in Spacecraft Rendezvous," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. PP, No. 99, 2020, pp. 1–9, 10.1109/tnnls.2020.3021037.

[15] C. W. T. Roscoe, J. J. Westphal, J. D. Griesbach, and H. Schaub, "Formation Establishment and Reconfiguration Using Differential Elements in J2-Perturbed Orbits," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 9, 2015, pp. 1–16, 10.2514/1.g000999.

[16] A. Richards, Y. Kuwata, and J. How, "Experimental Demonstrations of Real-time MILP Control," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003, 10.2514/6.2003-5802.

[17] B. Wu, D. Wang, E. K. Poh, and G. Xu, "Nonlinear Optimization of Low-Thrust Trajectory for Satellite Formation: Legendre Pseudospectral Approach," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 4, 2012, pp. 1371–1381, 10.2514/1.37675.

[18] D. Morgan, S.-J. Chung, and F. Y. Hadaegh, "Model Predictive Control of Swarms of Spacecraft Using Sequential Convex Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 1725–1740, 10.2514/1.g000218.

[19] B. A. Conway, *Spacecraft Trajectory Optimization*. Cambridge Aerospace Series, Cambridge University Press, 2010, 10.1017/cbo9780511778025.

[20] S. DiCairano, H. Park, and I. Kolmanovsky, "Model Predictive Control approach for guidance of spacecraft rendezvous and proximity maneuvering," *International Journal of Robust and Nonlinear Control*, Vol. 22, No. 12, 2012, pp. 1398–1427, 10.1002/rnc.2827.

[21] E. N. Hartley, P. A. Trodden, A. G. Richards, and J. M. Maciejowski, "Model predictive control system design and implementation for spacecraft rendezvous," *Control Engineering Practice*, Vol. 20, No. 7, 2012, pp. 695–713, 10.1016/j.conengprac.2012.03.009.

[22] A. W. Koenig, B. Macintosh, and S. D'Amico, "Formation Design of Distributed Telescopes in Earth Orbit for Astrophysics Applications," *Journal of Spacecraft and Rockets*, Vol. 56, No. 5, 2019, pp. 1–16, 10.2514/1.a34420.

[23] D. H. Jacobson and D. Q. Mayne, *Differential dynamic programming*. American Elsevier Pub. Co New York, 1970.

[24] D. M. Murray and S. J. Yakowitz, "Differential dynamic programming and Newton's method for discrete optimal control problems," *Journal of Optimization Theory and Applications*, Vol. 43, No. 3, 1984, pp. 395–414, 10.1007/bf00934463.

[25] J. Morimoto, G. Zeglin, and C. G. Atkeson, "Minimax differential dynamic programming: application to a biped walking robot," *SICE 2003 Annual Conference (IEEE Cat. No.03TH8734)*, Vol. 3, 2003, pp. 2310–2315 Vol.3.

[26] W. Sun, Y. Pan, J. Lim, E. A. Theodorou, and P. Tsiotras, "Min-Max Differential Dynamic Programming: Continuous and Discrete Time Formulations," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 12, 2018, pp. 2568–2580, 10.2514/1.g003516.

[27] Y. Tassa, T. Erez, and W. Smart, "Receding Horizon Differential Dynamic Programming," Vol. 20, 2007.

[28] Y. Aoyama, A. D. Saravanos, and E. A. Theodorou, "Receding Horizon Differential Dynamic Programming Under Parametric Uncertainty," *arXiv*, 2021.

[29] W. Li and E. Todorov, "Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems," *ICINCO (1)*, INSTICC Press, 2004, pp. 222–229.

[30] Z. Manchester and S. Kuindersma, "Derivative-Free Trajectory Optimization with Unscented Dynamic Programming," *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 3642–3647, 10.1109/cdc.2016.7798817.

[31] J. Rajamaki, K. Naderi, V. Kyrki, and P. Hämäläinen, "Sampled Differential Dynamic Programming," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1402–1409, 10.1109/iros.2016.7759229.

[32] T. A. Howell, B. E. Jackson, and Z. Manchester, "ALTRO: A Fast Solver for Constrained Trajectory Optimization," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7674–7679, 10.1109/iros40897.2019.8967788.

[33] Y. Aoyama, G. Boutselis, A. Patel, and E. A. Theodorou, "Constrained Differential Dynamic Programming Revisited," *arXiv*, 2020.

[34] Z. Xie, C. K. Liu, and K. Hauser, "Differential dynamic programming with nonlinear constraints," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 695–702, 10.1109/icra.2017.7989086.

[35] G. Lantoine and R. P. Russell, "A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 2: Application," *Journal of Optimization Theory and Applications*, Vol. 154, No. 2, 2012, pp. 418–442, 10.1007/s10957-012-0038-1.

[36] A. Pavlov, I. Shames, and C. Manzie, "Interior Point Differential Dynamic Programming," *IEEE Transactions on Control Systems Technology*, Vol. 29, No. 6, 2021, pp. 2720–2727, 10.1109/tcst.2021.3049416.

[37] G. Xu, F. Xiang, and Y. Chen, "Exact Dynamic of Satellite Relative Motion under Arbitrary Zonal Harmonic Perturbations," *2012 24th Chinese Control and Decision Conference (CCDC)*, Vol. 1, 2012, pp. 3676–3681, 10.1109/ccdc.2012.6244588.

[38] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and Stabilization of Complex Behaviors through Online Trajectory Optimization," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 4906–4913, 10.1109/iros.2012.6386025.

[39] R. Bonalli, T. Lew, and M. Pavone, "Analysis of Theoretical and Numerical Properties of Sequential Convex Programming for Continuous-Time Optimal Control," *arXiv*, 2020.

[40] J. Revels, M. Lubin, and T. Papamarkou, "Forward-Mode Automatic Differentiation in Julia," 2016.