Aerospace Science and Technology 136 (2023) 108216

Contents lists available at ScienceDirect

Aerospace Science and Technology

journal homepage: www.elsevier.com/locate/aescte

Trajectory planning for satellite cluster reconfigurations with sequential convex programming method

Lixiang Wang, Dong Ye *, Yan Xiao, Xianren Kong

Research Center of Satellite Technology, Harbin Institute of Technology, Harbin 150001, China

ARTICLE INFO

Article history:
Received 2 November 2022
Received in revised form 1 February 2023
Accepted 21 February 2023
Available online 5 March 2023
Communicated by Chaoyong Li

Keywords:
Large-scale satellite cluster
Sequential convex programming
Trajectory planning
Collision avoidance

ABSTRACT

This paper presents trajectory planning algorithms for large-scale satellite clusters reconfiguration based on sequential convex programming, and these algorithms consider fuel consumption and collision avoidance. Firstly, the trajectory planning problem is formulated as a nonconvex optimal control problem with nonlinear dynamics and nonconvex path constraints. Secondly, the original nonlinear continuous optimal control problem is transformed into a discrete convex optimization subproblem through linearization and discretization. Collision avoidance strategy between discrete points and obstacle avoidance constraints are considered in the convex subproblem to ensure that the trajectories are collision-free. Thirdly, coupled and decoupled sequential convex programming methods are proposed for rapidly generating collision-free and fuel-efficient trajectories. Finally, comparative numerical simulations are presented to demonstrate that as the number of satellites increases, 94 to 99 percent performance improvement over the pseudo-spectral method is achieved in terms of computational efficiency.

1. Introduction

Over the past few decades, much research effort has been devoted to satellite formations containing up to a dozen satellites [1,2]. Recently, large-scale satellite cluster consisting of numerous satellites has drawn considerable attention [3,4]. It is worth noting that large-scale satellite cluster refers to dozens or even hundreds of satellites rather than several satellites. The satellite cluster can achieve the complex functions of traditional large satellites with space networks, greatly reducing the costs and risks, and having good robustness and flexibility [5–7]. In order to realize diverse space missions and avoid space obstacles, satellite cluster reconfiguration is necessary. However, large-scale satellite cluster poses many difficulties to satellite cluster trajectory planning. Firstly, as the number of satellites increases, the probability of collision among satellites and between satellites and external obstacles becomes increasingly high. Secondly, due to the small weight of microsatellites, they have limited fuel, communication and computing capabilities. Therefore, computationally efficient trajectory planning algorithms are necessary for finding the minimum fuel-consuming trajectories and avoiding collisions in large-scale satellite cluster reconfiguration.

The goal of large-scale satellite cluster reconfiguration trajectory planning is to find a set of trajectories with minimum fuel consumption or minimum transfer time under collision avoidance constraints. It should be pointed out, however, the limits in fuel, communication and computational capabilities, perturbations in space, partial information of the environment and the large number of satellites bring trajectory planning more challenges and energy consumption. In recent years, a few researchers have investigated the reconfiguration of satellite clusters considering collision constraints [8,9]. In [10], the authors propose a self-organizing control method based on artificial potential functions, which can achieve cluster reconfiguration. Combining formation and satellite cluster, a finite-time distributed hierarchical control strategy is proposed to realize satellite cluster maintenance in [11]. For the optimal maneuvering of the satellite cluster, the effect of maneuvering time on the difference in mean semimajor axis among cluster members is studied in [12]. It has also been studied how to make the satellite clusters achieve periodic bounded relative motion via consensus [13,14]. It should be noted that, however, the above methods cannot be applied in satellite cluster reconfigurations in the presence of a large number of spacecrafts, the limited capability of individual microsatellites, obstacle avoidance, and minimum fuel consumption.

On the other hand, large-scale satellite cluster reconfiguration trajectory planning can be formulated as a set of optimal control problems with nonlinear dynamics and nonconvex state constraints [15,16]. There are many methods to solve these problems,

* Corresponding author.
E-mail address: yed@hit.edu.cn (D. Ye).

https://doi.org/10.1016/j.ast.2023.108216

including indirect and direct methods. The indirect method transforms the optimal problem into a Hamiltonian boundary value problem using the Pontryagin maximum principle [17,18]. Sandberg, et al. publish a comparison of autonomous spacecraft trajectory generation schemes including fuel-minimizing analytic optimization methods of Pontryagin and sinusoidal methods of deterministic artificial intelligence in [19]. Then, Raigoza, et al. [20] augment Sandberg's work with autonomous collision avoidance. Furthermore, Wilt, et al. [21] evaluate the efficacy of Sandberg and Raigoza's methods in the face of unaddressed parametric uncertainties. Due to the strong nonlinearity of the satellite cluster dynamic, it is difficult to solve the Hamiltonian boundary value problem. Different from indirect methods, direct methods convert the optimization problem into a finite-dimensional nonlinear programming problem by discretizing time, state and control variables. Direct methods mainly include the pseudospectral method [22], mixed-integer linear programming [23] and intelligent optimization methods [24]. However, it is worth mentioning that the methods mentioned above are sensitive to the increasing number of satellites because the number of constraints and variables is greatly increased.

Fortunately, as computer technology develops, convex optimization is adopted in a wide range of applications, where globally optimal solutions can be obtained in a finite time [25]. In [26], convex optimization is applied in trajectory planning for unpowered winged vehicles. A guidance algorithm is presented based on convex optimization for the generation of fuel-optimal 6-DOF spacecraft close proximity trajectory in [27]. In [28], the authors propose a method based on convex optimization to solve trajectory planning problems of vehicles with obstacle avoidance constraints. Co-planning scheme of motion and communication for team robots is developed by solving a single convex optimization problem in [29]. In addition, the authors adopt convex optimization to obtain fuel-efficient trajectories for multi-satellite formations [30]. It is a remarkable fact that, however, the convexification of nonconvex constraints leads to an overly conservative approximation of the feasible region, which is prone to infeasible solutions.

Sequential Convex Programming (SCP) is a local iterative optimization method. The nonlinear optimal control problem is approximated to a series of convex subproblems, and then SCP is employed to solve them. It can find feasible and efficient solutions in a short period. The application of SCP in aerospace engineering is in its infancy, but it is one of the candidate methods for real-time optimization [31,32]. Applications of SCP in the aerospace domain include Mars Entry [33], planet landing [34], launch vehicle [35], missile guidance [36] and UAVs trajectory planning [37,38]. Regarding satellite formation reconfiguration, the authors combine sequential convex programming with the genetic algorithm to propose a novel method for autonomous reconfiguration [39]. In addition, concerning satellite clusters, a guidance approach combining SCP and receding horizon control is proposed [40]. Then, in [41], the authors develop a decentralized scheme to handle autonomous trajectory planning for fractionated spacecraft reconfiguration. Nevertheless, in particular, SCP has not been applied much in large-scale satellite cluster reconfiguration and requires further development. At the same time, it is important to point out that collision avoidance among discrete points and external obstacles must be carefully considered when concerning hundreds of satellites.

The main contributions of this paper are as follows.

(1) The nonconvex optimal control problem for the large-scale satellite cluster reconfiguration is formulated considering J_2 perturbation. Then, the problem is transformed into convex forms by linearization and discretization.

(2) A collision avoidance strategy between discrete points is proposed, which not only ensures collision avoidance between adjacent discrete points but also prevents collisions between satellites and obstacles.

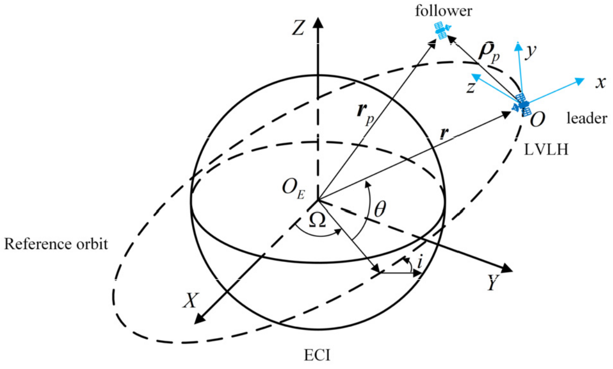


Fig. 1. Definition of coordinate frames.

(3) Due to a large number of constraints and the conservative convex approximation in solving the large-scale satellite cluster reconfiguration problem, the L1 penalty function is employed to prevent the SCP methods from falling into an infeasible solution space and failing to converge.

(4) Two sequential convex programming algorithms of coupling and decoupling are proposed for generating collision-free and fuel-efficient transfer trajectories throughout the large-scale satellite cluster reconfiguration. Compared with the GPOPS method, the computational efficiency is greatly improved while consuming similar fuel.

The paper is organized as follows. Section 2 describes the reconfiguration problem of large-scale satellite clusters as a nonconvex optimal control problem. Section 3 is the convexification of the nonconvex constraints. Coupled and decoupled sequential convex programming algorithms are developed in Section 4 and Section 5, respectively. In Section 6, comparative numerical simulations are conducted to demonstrate the effectiveness and efficiency of the two SCP methods. Section 7 concludes this paper.

2. Problem formulation

In this section, large-scale satellite cluster reconfiguration problems are described as the time-continuous, nonconvex optimization problems, including the relative dynamics, control inputs, objective function, and other constraints. It is worth mentioning that satellite cluster reconfiguration maneuvers can be attributed to mission changes or threats from unexpected external obstacles. There, satellites need to perform orbit transfers in time.

2.1. Nonlinear J_2 relative dynamics

To clearly describe the variables and constraints of the optimal problem, this paper uses two Cartesian coordinate systems, as shown in Fig. 1. Firstly, the Earth-Centered-Inertial (ECI) frame is adopted to locate the position of the leader satellite or virtual reference satellite. This frame is originated from the Earth's center, with the X axis pointing to the vernal equinox, the Z axis pointing to the North Pole, and the Y axis is defined by the right-hand rule. Secondly, the Local-Vertical-Local-Horizontal (LVLH) coordinate system is used to describe the relative position and velocity of the follower satellite with respect to the leader. The LVLH frame is located at the center of mass of the leader satellite, the x axis is directed from the center of the Earth to the leader, the z axis is perpendicular to the orbital plane of the leader and points to the direction of orbital angular momentum, and the y axis completes the right-handed coordinate system.

Then, the LVLH coordinate is spanned by the unit vectors

$$\hat{\boldsymbol{x}} = \boldsymbol{r}/r, \quad \hat{\boldsymbol{y}} = \hat{\boldsymbol{z}} \times \hat{\boldsymbol{x}}, \quad \hat{\boldsymbol{z}} = \boldsymbol{h}/h \tag{1}$$

The component of the angular velocity about the y-axis is zero [42], which means $\omega_y = 0$. The angular velocity of the rotating LVLH frame is

$$\boldsymbol{\omega} = \omega_x \hat{\boldsymbol{x}} + \omega_z \hat{\boldsymbol{z}} \tag{2}$$

where, $\boldsymbol{r}$ and $\dot{\boldsymbol{r}}$ denote the position and velocity vectors of the leader, respectively, $\boldsymbol{r}_p$ and $\dot{\boldsymbol{r}}_p$ refer to the position and velocity vectors of the follower, respectively, $\boldsymbol{h} = \boldsymbol{r} \times \dot{\boldsymbol{r}}$ represents the angular momentum vector per unit mass. Additionally, $r = |\boldsymbol{r}|$ and $h = |\boldsymbol{h}|$ identify the geocentric distance and angular momentum magnitude of the leader, respectively. The orbital elements used for the leader (reference) orbit are: geocentric distance ($\boldsymbol{r}$), radial velocity (v_x), orbital angular momentum (h), orbital inclination (i), right ascension of the ascending node (Ω), and argument of latitude (θ). This set of orbital parameters is able to completely determine the motion of the leader in the ECI coordinate system. Further, once the leader's state is known, the position and velocity vectors of the follower with respect to the leader can be expressed respectively as $\boldsymbol{\rho}_p = \begin{bmatrix} x_p & y_p & z_p \end{bmatrix}^{\mathrm{T}}$ and $\dot{\boldsymbol{\rho}}_p = \begin{bmatrix} \dot{x}_p & \dot{y}_p & \dot{z}_p \end{bmatrix}^{\mathrm{T}}$, where the superscript T represents the transpose of the vector. In addition, for the sake of concise expressions, this paper defines $s_\circ = \sin(\circ)$ and $c_\circ = \cos(\circ)$.

Then, it is assumed that all satellites are equipped with thrusters in each direction (x, y, z), providing continuous acceleration. Hence, the relative motion of the follower in the LVLH frame under the effect of the Earth's spherical gravity and J_2 perturbation can be described by [43].

$$\begin{cases} \ddot{x}_p = 2\dot{y}_p\omega_z - x_p(\eta_p^2 - \omega_z^2) - (\varsigma_p - \varsigma)s_i s_\theta + y_p\dot{\omega}_z \\ \quad -z_p\omega_x\omega_z - r(\eta_p^2 - \eta^2) + u_{px} \\ \ddot{y}_p = -2\dot{x}_p\omega_z + 2\dot{z}_p\omega_x - x_p\dot{\omega}_z + z_p\dot{\omega}_x - y_p(\eta_p^2 - \omega_z^2 - \omega_x^2) \\ \quad -(\varsigma_p - \varsigma)s_i c_\theta + u_{py} \\ \ddot{z}_p = -2\dot{y}_p\omega_x - x_p\omega_x\omega_z - y_p\dot{\omega}_x - z_p(\eta_p^2 - \omega_x^2) - (\varsigma_p - \varsigma)c_i \\ \quad +u_{pz} \end{cases} \tag{3}$$

where u_{px}, u_{py}, u_{pz} are the thrust acceleration on the follower satellite. η, η_p, ξ, ξ_p, r_p, r_{pZ} are to simplify the equations, and they can be obtained from

$$\begin{cases} \varsigma = \frac{2k_{J2}s_i s_\theta}{r^4} \\ \varsigma_p = \frac{2k_{J2}r_{pZ}}{r_p^5} \\ \eta^2 = \frac{\mu}{r^3} + \frac{k_{J2}}{r^5} - \frac{5k_{J2}s_i^2 s_\theta^2}{r^5} \\ \eta_p^2 = \frac{\mu}{r_p^3} + \frac{k_{J2}}{r_p^5} - \frac{5k_{J2}r_{pZ}^2}{r_p^7} \\ r_p = \sqrt{\left(r + x_p\right)^2 + y_p^2 + z_p^2} \\ r_{pZ} = \left(r + x_p\right) s_i s_\theta + y_p s_i s_\theta + z_p c_i \end{cases} \tag{4}$$

where $k_{J2} = \frac{3}{2} J_2 \mu R_e^2$, J_2 denotes the second zonal harmonic coefficient of the Earth, μ represents the Earth's gravitational constant, and R_e is the Earth's equatorial radius.

Under the effect of the J_2 perturbation, the leader satellite's dynamics equations are as follows:

$$\begin{cases} \dot{r} = v_x \\ \dot{v}_x = -\frac{\mu}{r^2} + \frac{h^2}{r^3} - \frac{k_{J2}}{r^4}(1 - 3s_i^2 s_\theta^2) \\ \dot{h} = -\frac{k_{J2}s_i^2 s_{2\theta}}{r^3} \\ \dot{\Omega} = -\frac{2k_{J2}c_i s_\theta^2}{hr^3} \\ \dot{i} = -\frac{k_{J2}s_{2i}s_{2\theta}}{2hr^3} \\ \dot{\theta} = \frac{h}{r^2} + \frac{2k_{J2}c_i^2 s_\theta^2}{hr^3} \end{cases} \tag{5}$$

In summary, the relative dynamics of the leader and the follower under J_2 disturbance have been formulated.

2.2. Control input and objective function

In order to reduce the use of subscripts and to make the presentation of this paper more concise, $\|\boldsymbol{x}\|_\alpha$ is defined here. $\boldsymbol{x} = (x_1, x_2, ..., x_n)^{\mathrm{T}}$ is the vector. $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$ represents the sum of the absolute values of the elements of the vector. $\|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + \cdots + x_n^2}$ is the Euclidean norm. $\|\boldsymbol{x}\|_\infty = \max |x_i|$ means the maximum norm.

Assume that the large-scale satellite cluster consists of M almost identical satellites, meaning that they all have one thruster in each direction and each thruster has the same upper limit. In the LVLH coordinate system, $\boldsymbol{u}_p(t) = [u_{px}(t) u_{py}(t) u_{pz}(t)]^{\mathrm{T}}$ is specified as the control variable. And the thrust constraint is expressed as

$$\| \boldsymbol{u}_p \|_\infty \le T_{\max}, \quad p = 1, \ldots, M \tag{6}$$

where $T_{\max}$ is the magnitude of the maximum thrust of each thruster.

Remark 1 *(optimization objective).* The optimal problem needs to define its optimization objective. Even if the satellite cluster contains hundreds of satellites and there are some obstacles in space, sometimes we can still find many feasible trajectories that satisfy the collision avoidance constraint. Therefore, evaluating the merits of these trajectories is an important issue. For instance, when satellites suddenly encounter some external obstacles, they are expected to be able to perform maneuvers to avoid these obstacles in a very short period. Typically, however, to prolong the satellite's lifetime, we expect the satellite to consume as less fuel as possible during the whole transfer process. Therefore, in this paper, fuel minimization is the objective of the large-scale satellite cluster reconfiguration maneuver problem.

The objective function is as follows.

$$\min_{\boldsymbol{u}_p, p=1,\ldots,M} \sum_{p=1}^{M} \int_0^{t_f} \| \boldsymbol{u}_p \|_1 \, \mathrm{d}t \tag{7}$$

where t_f is the mission time of the entire orbital transfer. The total fuel consumption is equal to the sum of the fuel consumed in each direction.

2.3. Constraints

The constraints in the optimal control problem for large-scale cluster reconfiguration include not only the control magnitude in Eq. (6), but also the dynamics in Eq. (8), inter-satellite collision avoidance in Eq. (10), and boundary conditions in Eq. (12). Moreover, due to the various external obstacles such as space debris, obstacle avoidance needs to be considered in Eq. (11). It is important to point out that the shape of the obstacles is assumed to be spherical and static in the LVLH coordinate system.

$$\dot{\boldsymbol{y}}_p = f\left(\boldsymbol{y}_p, \boldsymbol{o}\right) + \boldsymbol{B}\boldsymbol{u}_p, \quad p = 1, \ldots, M \tag{8}$$

$$\dot{\boldsymbol{o}}(t) = f_{\text{leader}}(\boldsymbol{o}(t)) \tag{9}$$

$$\| \boldsymbol{G}(\boldsymbol{y}_p - \boldsymbol{y}_q) \|_2 - d_{\min} \geq 0, \quad q < p, \quad p = 2, \ldots, M \tag{10}$$

$$\| \boldsymbol{G}\boldsymbol{y}_p - \mathbf{O}_i \|_2 - d_{\text{obs},i} \geq 0, \quad p = 1, \ldots, M, \quad \mathbf{O}_i \in \mathrm{O} \tag{11}$$

$$\boldsymbol{y}_p(0) = \boldsymbol{y}_{p,0}, \quad \boldsymbol{y}_p\left(t_{\text{f}}\right) = \boldsymbol{y}_{p,\text{f}}, \quad p = 1, \ldots, M \tag{12}$$

where Eq. (8) and Eq (9) are rewritten from Eq. (3) and Eq (5), respectively, and Eq. (9) describes the motion of the leader (reference satellite) in the ECI coordinate system. $\boldsymbol{y}_p = (\boldsymbol{\rho}_p^T, \dot{\boldsymbol{\rho}}_p^T)^{\text{T}}$ denotes the state of the follower satellite p relative to the reference satellite. The state of the reference satellite is expressed by $\boldsymbol{o} = [r, v_x, h, i, \Omega, \theta]^{\text{T}}$. Assuming that the leader is not equipped with thrusters, i.e., unpowered flight, and the initial state is known, the orbital parameters of the leader at any subsequent time can be calculated according to the differential equation Eq. (9). $d_{\min}$ identifies the minimum safe distance between any two satellites, O specifies the set of states related to all obstacles, $\boldsymbol{O}_i$ represents the ith obstacle, $d_{\text{obs},i}$ denotes the minimum allowable distance between satellites and the ith obstacle, $\boldsymbol{y}_{p,0}$ and $\boldsymbol{y}_{p,\text{f}}$ denote the initial and terminal states, respectively, $\boldsymbol{B} = \begin{bmatrix} 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix}^{\text{T}}$, $\boldsymbol{G} = \begin{bmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \end{bmatrix}$.

2.4. Nonconvex optimal control problem

Finally, the satellite cluster reconfiguration problem is transformed into a nonconvex optimal control problem. The objective of this optimization problem is to find a set of fuel-optimal trajectories from the initial states to the terminal, while all constraints need to be satisfied. Therefore, the satellite cluster reconfiguration can be defined as Problem 1 (P1). P1 is an optimization problem with nonconvex state constraints.

Problem 1 *(nonconvex optimization problem).*

$$\begin{cases} \min_{\boldsymbol{u}_p, p=1,\ldots,M} \sum_{p=1}^{M} \int_0^{t_{\text{f}}} \| \boldsymbol{u}_p \|_1 \, \mathrm{d}t \\ \text{subject to Eqs. (6), (8), (10), (11), (12)} \end{cases} \tag{13}$$

3. Convexification of the nonconvex optimal control problem

Problem 1 has described the large-scale satellite cluster reconfiguration as a nonconvex optimal control problem. Nevertheless, since the satellites in the cluster tend to be small and have limited computational and communication capabilities, convex programming methods are used to approximate the problem in this paper. Then, SCP is adopted to solve a series of convex subproblems iteratively.

In this section, the nonconvex terms in P1 need to be convexified to construct the convex programming subproblem. It is worth pointing out that convexification of the dynamics, inter-satellite collision avoidance, and obstacle avoidance in Problem 1 is required, while the objective function, control magnitude constraints, and boundary condition constraints have met the requirements of convex programming problems.

3.1. Linearization and discretization of dynamics

The dynamics must be linearized and discretized to satisfy the form of the convex programming problem. The discretization yields a number of equations that can be solved by numerical methods. After linearization, the dynamics is rewritten as

$$\dot{\boldsymbol{y}}_p = \boldsymbol{A}\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right)\boldsymbol{y}_p + \boldsymbol{B}\boldsymbol{u}_p + \boldsymbol{s}\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right) \tag{14}$$

where $\bar{\boldsymbol{y}}_p$ is the reference trajectory in the linearization, $\boldsymbol{A}\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right)$ and $\boldsymbol{s}\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right)$ can be obtained by Eq. (15).

$$\boldsymbol{A}\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right) = \left.\frac{\partial f}{\partial \boldsymbol{y}_p}\right|_{\bar{\boldsymbol{y}}_p}, \boldsymbol{s}\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right) = f\left(\bar{\boldsymbol{y}}_p, \boldsymbol{o}\right) - \left.\frac{\partial f}{\partial \boldsymbol{y}_p}\right|_{\bar{\boldsymbol{y}}_p} \bar{\boldsymbol{y}}_p \tag{15}$$

After linearization, the dynamics needs to be discretized using the zero-order hold method, and the total reconfiguration transfer time is equidistantly discretized and partitioned into K intervals, resulting in $K+1$ discrete points. t_0 and t_{f} specify the start and the end of the mission, respectively. Then, the time step is denoted as $\Delta t = (t_{\text{f}} - t_0)/K$. The control inputs are considered to change only at the sample moment and to remain constant between two adjacent discrete points.

$$\boldsymbol{u}_p(t) = \boldsymbol{u}_p[k], \quad t \in [t_k, t_{k+1}), \quad k = 0, 1, ..., K-1 \tag{16}$$

After discretization, the dynamics is rewritten as

$$\begin{aligned} &\boldsymbol{y}_p[k+1] = \boldsymbol{A}_p[k]\boldsymbol{y}_p[k] + \boldsymbol{B}_p[k]\boldsymbol{u}_p[k] + \boldsymbol{s}_p[k], \\ &k = 0, \ldots, K-1, \quad p = 1, \ldots, M \end{aligned} \tag{17}$$

where

$$\begin{cases} \boldsymbol{y}_p[k] = \boldsymbol{y}_p\left(t_k\right) \\ \boldsymbol{u}_p[k] = \boldsymbol{u}_p\left(t_k\right) \\ \boldsymbol{o}[k] = \boldsymbol{o}(t_k) \\ \boldsymbol{A}_p[k] = e^{\boldsymbol{A}\left(\bar{\boldsymbol{y}}_p(t_k), \boldsymbol{o}(t_k)\right)\Delta t} \\ \boldsymbol{B}_p[k] = \int_0^{\Delta t} e^{\boldsymbol{A}\left(\bar{\boldsymbol{y}}_p(t_k), \boldsymbol{o}(t_k)\right)\xi} \boldsymbol{B} \mathrm{d}\xi \\ \boldsymbol{s}_p[k] = \int_0^{\Delta t} e^{\boldsymbol{A}\left(\bar{\boldsymbol{y}}_p(t_k), \boldsymbol{o}(t_k)\right)\xi} \boldsymbol{s}\left(\bar{\boldsymbol{y}}_p(t_k), \boldsymbol{o}(t_k)\right) \mathrm{d}\xi \end{cases} \tag{18}$$

So far, the nonlinear dynamics in P1 is transformed into linear constraint. In addition, the remaining constraints are discretized in the following forms.

$$\| \boldsymbol{u}_p[k] \|_\infty \leq T_{\max}, \quad k = 0, \ldots, K-1, \quad p = 1, \ldots, M \tag{19}$$

$$\begin{aligned} &\| \boldsymbol{G}\left(\boldsymbol{y}_p[k] - \boldsymbol{y}_q[k]\right) \|_2 - d_{\min} \geq 0, \quad k = 0, \ldots, K, \\ &q < p, \quad p = 2, \ldots, M \end{aligned} \tag{20}$$

$$\begin{aligned} &\| \boldsymbol{G}\boldsymbol{y}_p[k] - \mathbf{O}_i \|_2 - d_{\text{obs},i} \geq 0, \quad k = 0, \ldots, K, \\ &p = 1, \ldots, M, \quad \mathbf{O}_i \in \mathrm{O} \end{aligned} \tag{21}$$

$$\boldsymbol{y}_p[0] = \boldsymbol{y}_{p,0}, \quad \boldsymbol{y}_p[K] = \boldsymbol{y}_{p,\text{f}}, \quad p = 1, \ldots, M \tag{22}$$

3.2. Convexification of inter-satellite collision avoidance constraint

From Eq. (20), it can be seen that the inter-satellite collision avoidance constraint is concave. Hence, this constraint needs to be convexified to make it an acceptable form for the convex programming problem. Next, how to convexify the inter-satellite collision avoidance constraints will be explained with geometric figures. As shown in Fig. 2, (a) represents the real no-fly zone of inter-satellite collision avoidance constraint. The zone is actually a sphere in 3-D with the center being the center of mass of the satellite and the radius being the minimum safe distance. (b) describes the convexification process. Firstly, the line segment l_{qp} connects the centers of mass of any two satellites, and then the line l_p is perpendicular to l_{qp}. The line l_p (or plane in the 3-d view) divides the entire space into two parts, with the lower left half-plane being the newly defined forbidden area. It is worth mentioning that, on the one hand, this area contains the initially defined forbidden region. On the other hand, it is a convex set. This means that the

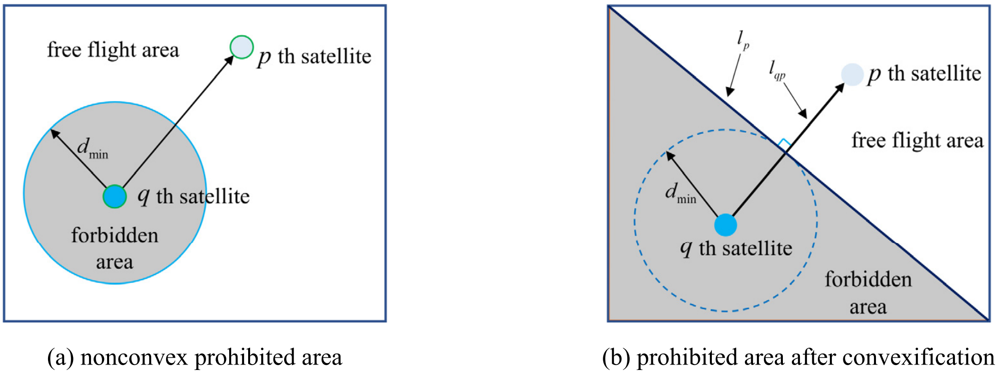


Fig. 2. Convexification of inter-satellite collision avoidance.

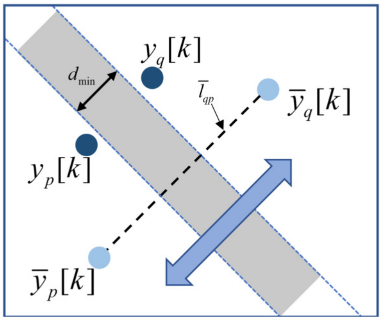


Fig. 3. Schematic diagram of collision avoidance constraints between satellites.

new prohibited area is not only collision-proof for satellites, but also acceptable for the convex programming problem. As shown in Fig. 3, the exact collision avoidance constraint only requires that the distance between two satellites is not less than the safety distance, while the convexified constraint becomes that two satellites cannot be located in the band of width $d_{\min}$ at the same time. The extension of the band is perpendicular to the dashed line $\bar{l}_{qp}$, which connects the discrete points of the nominal trajectories of the two satellites. In particular, the position of this band is not fixed and will change with the reference points.

Hence, the convexified inter-satellite collision avoidance constraint is as follows

$$\boldsymbol{E}^{\mathrm{T}}\boldsymbol{G}\left(\boldsymbol{y}_p[k]-\boldsymbol{y}_q[k]\right)-d_{\min}\parallel \boldsymbol{E}\parallel_2\geq 0,\quad k=0,\ldots,K-1,$$
$$q<p,\quad p=2,\ldots,M \tag{23}$$

where $\mathbf{E}=\boldsymbol{G}\left(\bar{\boldsymbol{y}}_p[k]-\bar{\boldsymbol{y}}_q[k]\right)$, $\bar{\boldsymbol{y}}_p$ and $\bar{\boldsymbol{y}}_q$ represent the nominal trajectories of satellites p and q, respectively. Now, the convexified collision avoidance constraint is affine and satisfies the convex programming form.

Remark 2 *(nominal trajectories).* The nominal trajectory $\bar{\boldsymbol{y}}_p$ is the result of the last iteration of the true trajectory $\boldsymbol{y}_p$. Collision avoidance constraints are convexified by the nominal trajectory. Therefore, these nominal trajectories are known. It is important to note that the above approximate convexification is accurate only when the deviation between the nominal and actual trajectory is quite small. The nominal trajectory will be further explained in Section 4.1.

3.3. Convexification of obstacle avoidance constraint

Next, the obstacle avoidance constraint needs to be convexified since it is concave. As already mentioned above, the obstacle is assumed to be spherical and static in the LVLH coordinate system. Eq. (21) is rewritten at the corresponding discrete point $\bar{\boldsymbol{y}}_p[k]$ of the nominal trajectory and it can be transformed into the following affine function.

$$(\boldsymbol{G}\boldsymbol{y}_p[k]-\mathbf{O}_i)^{\mathrm{T}}(\boldsymbol{G}\bar{\boldsymbol{y}}_p[k]-\mathbf{O}_i)-d_{\mathrm{obs},i}\left\|(\boldsymbol{G}\bar{\boldsymbol{y}}_p[k]-\mathbf{O}_i)\right\|_2\geq 0,$$
$$k=0,\ldots,K,\quad p=1,\ldots,M,\quad \mathbf{O}_i\in\mathrm{O} \tag{24}$$

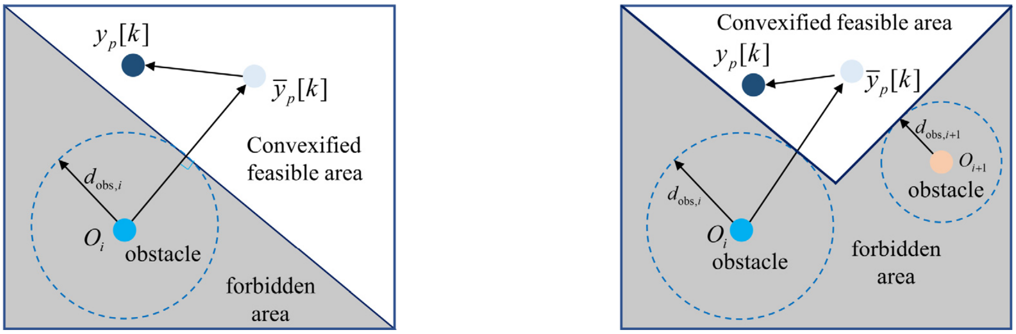
Similarly, the convexification of the spherical obstacle avoidance constraint can be understood intuitively with geometric figures in the same way as the inter-satellite collision avoidance constraint. As shown in Fig. 4, (a) indicates how the obstacle collision avoidance constraints are convexified at the discrete points. (b) demonstrates the case of avoiding multiple obstacles, and the convexified feasible domain is the intersection of two half-planes, which is still a convex set.

3.4. Obstacle avoidance constraint in discrete intervals

According to Eq. (24), the obstacle avoidance constraint only acts at the discrete points and does not guarantee that the trajectory between two adjacent discrete points is collision-free. Although the immediate thought is to reduce the time step as much as possible to reduce the possibility of collisions, this will inevitably lead to an increase in variables and constraints. In addition, the size of the original convex subproblem will increase significantly, further leading to a decrease in computational efficiency or failure to find a valid solution. Therefore, to improve the safety margin within the discrete interval, the proposed approach in this paper is to add the following constraint between discrete points.

$$(\boldsymbol{G}\boldsymbol{y}_p[k]-\mathbf{O}_i)^{\mathrm{T}}(\boldsymbol{G}\bar{\boldsymbol{y}}_p[k-1]-\mathbf{O}_i)-d_{\mathrm{obs},i}\left\|(\boldsymbol{G}\bar{\boldsymbol{y}}_p k-1]-\mathbf{O}_i)\right\|_2\geq 0,$$
$$k=1,\ldots,K,\quad p=1,\ldots,M,\quad \mathbf{O}_i\in\mathbf{O} \tag{25}$$

Fig. 5 illustrates the geometric principle of the obstacle avoidance method between discrete points. From the previous section, a half-plane without an intersection with the obstacle can be determined based on the obstacle and a discrete point $\bar{\boldsymbol{y}}_p[k]$ of the nominal trajectory. This half-plane is the convexified feasible domain of $\boldsymbol{y}_p[k]$. Correspondingly, such a half-plane feasible domain exists for $\boldsymbol{y}_p[k-1]$. Suppose the feasible domain of $\boldsymbol{y}_p[k]$ is constrained to be the intersection of the above two half-planes (the feasible domain on the right of the Fig. 5). In that case, $\boldsymbol{y}_p[k]$ is still in the same half-plane feasible domain as $\boldsymbol{y}_p[k-1]$ while satisfying its obstacle avoidance constraint. Since the half-plane is a convex set, according to the definition of a convex set, the line segment between any two points in the convex set lies within the convex set. Then, if the discrete points $\boldsymbol{y}_p[k]$ and $\boldsymbol{y}_p[k-1]$ are in the convex feasible domain, the line segment connecting these



(a) Convexification of single obstacle avoidance

(b) Convexification of multiple obstacle avoidance

Fig. 4. Convexification of obstacle avoidance constraint.

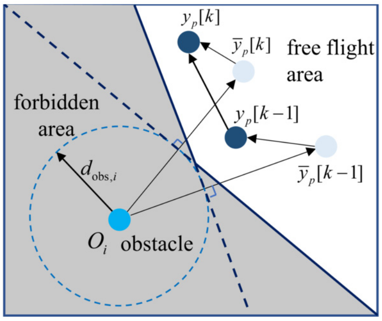


Fig. 5. Convexification of obstacle collision avoidance constraint between discrete points.

two points will not intersect the obstacle spheres. It means that the strategy can guarantee that straight trajectories between adjacent discrete points will not collide with obstacles and reduce the possibility of collisions.

4. Coupled sequential convex programming method

In this section, SCP is adopted to tackle the modified large-scale cluster reconfiguration maneuver problem. It is based on convex programming, where the almost optimal trajectories are found successively by iteratively solving convex subproblems. It is worth mentioning that the accuracy of the solution of SCP is independent of the initial guess. Even if the initial solution is poor, it can still find the optimized solution through successive iterations. The original nonlinear optimization problem is first converted into the convex optimization problem. Then, the convex subproblem is constructed based on the L1 penalty function. Furthermore, the coupled SCP method is presented to solve this convex subproblem to obtain a series of fuel-efficient trajectories that satisfy the collision avoidance constraints.

4.1. Penalty function-based convex subproblem modeling

It should be pointed out that all non-convex constraints in P1 have been converted into convex forms by linearization and discretization. Therefore, this section rewrites the original nonlinear problem (P1) to yield a new convex optimization problem P2, which is shown below

Problem 2 *(convex problem).*

$$\begin{cases} \min_{\mathbf{u}_p, p=1,\dots,M} \sum_{p=1}^{M} \sum_{k=0}^{K-1} \parallel \boldsymbol{u}_p(k) \parallel_1 \Delta t \\ \text{subject to Eqs. (17), (19), (22)–(25)} \end{cases} \tag{26}$$

where the convexified dynamic (Eq. (17)) and collision avoidance constraints between satellites (Eq. (23)) correspond to Eq. (8) and Eq. (10) in Problem 1, respectively. And the constraint between satellite and obstacles in Eq. (11) is rewritten by Eq. (24). It is also important to note that Problem 2 adds the obstacle avoidance constraint in Eq. (25) between adjacent discrete points.

The nominal trajectories are used to approximately convexify the dynamics and collision avoidance constraints and they are important in solving the optimal problem of satellite cluster reconfiguration. In other words, in order to find a valid result of the optimization subproblem, the error between the nominal and true trajectories must be as small as possible. To ensure that the estimation accuracy of the nominal trajectory to the actual trajectory is high, SCP is adopted. SCP is an approach for solving nonconvex optimization problems based on convex programming, and it solves approximate convex subproblems by multiple iterations. The nominal trajectory of the first iteration is the given initial guess $\bar{\boldsymbol{Y}}_0$, and then $\bar{\boldsymbol{Y}}_i$ in the subsequent i-th iteration is the solution of the previous iteration $\boldsymbol{Y}_{i-1}$, which means $\bar{\boldsymbol{Y}}_i = \boldsymbol{Y}_{i-1}$. By the way, the iteration does not end until the convergence conditions are satisfied.

Therefore, there are two keys to the SCP approach. Firstly, the nominal trajectory should be close enough to the actual trajectory to ensure that the dynamics and collision avoidance constraints are linearized with high accuracy. Secondly, handling equality and inequality constraints to make it an unconstrained optimization problem to reduce the possibility of infeasible solutions. Concerning the first issue, the following trust region constraint is adopted at the current iteration to guarantee that the convexification is accurate and the successive iterations are convergent.

$$\begin{aligned} &\left\| \boldsymbol{y}_{p,i}[k] - \bar{\boldsymbol{y}}_{p,i}[k] \right\|_\infty \le \delta_i, \quad k = 0, \dots, K-1, \\ &p = 1, \dots, M \end{aligned} \tag{27}$$

where: δ_i is the radius of the trust region at the i-th iteration, $\delta_i = \eta\delta_{i-1}$, η identifies the trust region shrinkage factor which is a constant between 0 and 1. Apparently, the radius of the trust domain gradually shrinks with the increasing number of iterations, and therefore the final convergence can be guaranteed. It is worth mentioning that Eq. (27) is an affine function of the states and does not affect the convexity of the convex subproblem. Now, the convex subproblem (P3) with the trust region constraint is formulated as

Problem 3.

$$\begin{cases} \min_{\mathbf{u}_p, p=1,\dots,M} \sum_{p=1}^{M} \sum_{k=0}^{K-1} \parallel \boldsymbol{u}_p(k) \parallel_1 \Delta t \\ \text{subject to} \\ h_l(\boldsymbol{Y})=0, l=1,\dots,n_{\text{eq}} \\ g_l(\boldsymbol{Y}) \le 0, l=1,\dots,n_{\text{ineq}} \\ \left\| \boldsymbol{y}_{p,i}[k] - \bar{\boldsymbol{y}}_{p,i}[k] \right\|_\infty \le \delta_i,\ k=0,\dots,K-1,\ p=1,\dots,M \end{cases} \tag{28}$$

where $h_l(\boldsymbol{Y})=0, l=1,\dots,n_{\text{eq}}$ represents the equality constraints, including dynamic in Eq. (17) and boundary condition constraints in Eq. (22), $g_l(\boldsymbol{Y}) \le 0, l=1,\dots,n_{\text{ineq}}$ denotes the inequality constraints, including the thrust magnitude constraint in Eq. (19) and the collision avoidance constraints in Eqs. (23)–(25). The optimization variable $\boldsymbol{Y}$ represents the state of all satellites at the discrete points, as follows

$$\begin{cases} \boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_M)^{\text{T}} \\ \boldsymbol{y}_p = (\boldsymbol{y}_p[0], \dots, \boldsymbol{y}_p[K]), p=1,\dots,M \end{cases} \tag{29}$$

It is important to note that a large number of constraints and the conservative convex approximation are two challenges in solving the large-scale satellite cluster reconfiguration problem. It may result in cases where the subproblem does not have a feasible solution, especially for problems with terminal equality constraints. Although SCP can recover from infeasible solutions through a series of iterations, there are possibilities where it falls into an infeasible solution space and fails to converge. Currently, for solving such convex optimization problems, there is well-developed software, (for example, CVX). However, CVX cannot return infeasible solutions but invalid data when encountering infeasible convex subproblems. As a result, SCP is unlikely to run the next iteration and recover from an infeasible solution.

Considering the above issues, the L1 penalty function is employed. The equality and inequality constraints in Problem 3 are added to the objective function as penalty terms. The trust region constraint is used as a boundary condition constraint for the design variables. Then, the coupled convex subproblem (P4) can be constructed.

Problem 4 *(coupled convex problem).*

$$\begin{cases} \min_{\mathbf{u}_p, p=1,\dots,M} \sum_{p=1}^{M} \sum_{k=0}^{K-1} \parallel \boldsymbol{u}_p(k) \parallel_1 \Delta t \\ \quad +\omega_{\text{eq}} \sum_{l=1}^{n_{\text{eq}}} |h_l(\boldsymbol{Y})| + \omega_{\text{ineq}} \sum_{l=1}^{n_{\text{ineq}}} |g_l(\boldsymbol{Y})|^+ \\ \text{subject to} \\ \left\| \boldsymbol{y}_{p,i}[k] - \bar{\boldsymbol{y}}_{p,i}[k] \right\|_\infty \le \delta_i,\ k=0,\dots,K-1,\ p=1,\dots,M \end{cases} \tag{30}$$

where ω_{eq} and ω_{ineq} refer to the penalty coefficients of the equality constraints and inequality constraints, respectively, $|g_l(\boldsymbol{Y})|^+ = \max(g_l(\boldsymbol{Y}), 0)$, $|h_l(\boldsymbol{Y})|$ indicates the absolute value of $h_l(\boldsymbol{Y})$.

Remark 3 *(L1 penalty function).* The L1 penalty function is known as the exact penalty method [44]. That means if the penalty term is multiplied by a large coefficient, then the minimum objective of the convex subproblem P4 is almost equal to the minimum objective of Problem 3. L1 penalty method is a considerably stable numerical algorithm that enables the penalty terms to eventually equal 0, that is, the constraints are satisfied.

4.2. Coupled SCP -based satellite cluster reconfiguration

Compared to the single convex programming method, sequential convex programming reduces the dependence of solutions on initial values by sequentially solving convex approximation problems corresponding to different nominal trajectories. The specific steps of coupled SCP are shown in Algorithm 1.

Algorithm 1 Coupled SCP-based Satellite Cluster Reconfiguration.

```
Initialization: $\bar{\boldsymbol{Y}} = \boldsymbol{Y}_0, \delta_i = \delta_0, i = 0, \eta$
1  while $\boldsymbol{Y}_i$ does not satisfy Eqs. (17), (19), (22)
2      $i = i + 1, \bar{\boldsymbol{Y}} = \boldsymbol{Y}_{i-1}, \delta_i = \eta \bullet \delta_{i-1}$
3      $\boldsymbol{Y}_i \leftarrow$ the solution to Problem 4 without Eqs. (23)–(25)
4  end while
5  $\delta_i = \delta_0$
6  while $\boldsymbol{Y}_i$ does not satisfy Eqs. (17), (19), (22)–(25), (31) and (32)
7      $i = i + 1, \bar{\boldsymbol{Y}} = \boldsymbol{Y}_{i-1}, \delta_i = \eta \bullet \delta_{i-1}$
8      $\boldsymbol{Y}_i \leftarrow$ the solution to Problem 4
9  end while
10 Return $\boldsymbol{Y}_i$
```

To decrease the sensitivity to the initial guess, the whole algorithm contains two phases. In this algorithm, $\bar{\boldsymbol{Y}}$ denotes the reference trajectory. $\boldsymbol{Y}_0$ refers to the initial guess. δ_0 identifies the initial trust domain. i is the number of iterations. η specifies the trust region shrinkage factor.

First, initializing parameters, which are the initial nominal trajectories of all satellites $\boldsymbol{Y}_0$, the initial trust domain δ_0, and the trust region shrinkage factor η. Then, in phase 1 (line 1-4), the convex subproblem P4 is resolved iteratively, but the collision avoidance constraints (including inter-satellite collision avoidance and obstacle avoidance constraints) are not considered at this moment. The iteration terminates when the trajectory satisfies the dynamics, control magnitude constraints and boundary conditions, and this result is employed as the nominal trajectory for the next stage.

It should be pointed out that the trust domain should be reset to the initial domain before the second phase starts. Then, in phase 2 (line 6-9), the complete convex problem P4 containing all constraints is solved iteratively. At each iteration, the planned trajectory results of the last iteration are regarded as nominal solutions for the current iteration. Meanwhile, the trust domain is shrinking with increasing number of iterations. Finally, the iteration does not stop until the convergence conditions Eq. (31) and Eq. (32) are satisfied. $\boldsymbol{Y}_i$ is regarded as the final solution for all satellites.

$$\left\| \boldsymbol{y}_{p,i}[k] - \boldsymbol{y}_{p,i-1}[k] \right\|_\infty \le \varepsilon, \quad k=0,\dots,K, \quad p=1,\dots,M \tag{31}$$

$$\left\| \boldsymbol{G}\left(\boldsymbol{y}_{p,i}[k] - \boldsymbol{y}_{q,i}[k] \right) \right\|_2 \ge d_{\min}, \quad k=0,\dots,K, \\ q<p, \quad p=2,\dots,M \tag{32}$$

where $\boldsymbol{y}_{p,i}[k]$ refers to the state of the p-th satellite, at the k-th discrete point, after the i-th iteration. Eq. (31) means that the variation of the results of two consecutive iterations needs to be less than the tolerance ε, and Eq. (32) represents the actual collision constraint of any two satellites. Finally, phase 2 ends once the two convergence conditions are satisfied and the approximate optimal solutions of the original problem 1 are obtained.

Coupled SCP method is effective for a satellite cluster consisting of dozens of satellites. Compared with the initial nonlinear optimal problem P1, the coupled SCP method transforms the problem into a convex form, thus greatly increasing the possibility of the solution and the computational speed. However, the coupled method has two drawbacks. First, this method calculates the collision-free trajectories for all satellites simultaneously, which imposes a huge computational and communication burden on the leader satellite. Second, the inter-satellite collision avoidance constraints are coupled. The number of these constraints grows quadratically as the number of satellites increases. Therefore, finding a distributed SCP method is necessary to reduce the size and computation time of the optimization problem.

5. Decoupled sequential convex programming

As discussed above, the coupled SCP can generate fuel-saving trajectories for dozens of satellites. However, as shown in Eq. (23), both satellites p and q are unknown. If satellite p wants to avoid satellite q, then the state of satellite q at the same time needs to be known. Similarly, the prerequisite for satellite q to avoid satellite p is that the state of satellite p needs to be known. That is, for the satellites in the cluster, their states need to be computed simultaneously to satisfy the collision constraint. In summary, the collision avoidance constraint Eq. (23) leads to the coupling of the states of all satellites, which requires a particular leader satellite to plan the trajectories for all satellites. The problem complexity rises significantly with the number of satellites. Moreover, satellites in the cluster are usually small in size and have limited computing and communication capabilities. It is quite difficult to find a satellite to perform such a complex, large-scale computation. Even if there is a satellite with high computing power, then the whole mission will fail once that satellite breaks down, resulting in poor robustness of coupled SCP.

Therefore, considering practical scenarios and robustness, this section proposes a decoupled sequential convex programming method based on coupled SCP by decoupling the inter-satellite collision avoidance constraints. As shown in Eq. (33), only $\boldsymbol{y}_p$ is unknown and the nominal trajectory $\bar{\boldsymbol{y}}_q$ is the trajectory obtained from the previous iteration of satellite p, which is known. The reason why the nominal trajectory can replace the actual trajectory will be elaborated later. Therefore, satellite p does not need to know the current state of other satellites. The state between satellites is independent, that is, decoupled. Each satellite can compute its own trajectory independently. Therefore, at each iteration, we can decompose the coupled convex problem (P4) into M small independent problems for supporting parallel computing. This further improves the computational efficiency of the algorithm and enhances its scalability to the number of satellites.

In particular, the coupled SCP is that the leader satellite provides transfer plans for all satellites, while the decoupled SCP allows each satellite to independently determine its control inputs by decoupling the inter-satellite collision avoidance constraints.

5.1. Decoupled convex-programming subproblem

According to the analysis of problem 4, the inter-satellite collision avoidance constraint depends on other satellites' states, and each satellite must know the trajectories of the other spacecraft. This means solving the optimization problem simultaneously, which is apparently inappropriate for large-scale satellite clusters. In order to decouple the collision constraints, the constraints are reorganized.

First, the decoupling of collision constraints between satellites is achieved based on the idea of trajectory freezing. With increasing number of iterations, the trajectory of the current iteration becomes more and more similar to the trajectory of the previous iteration. Thus, for the p-th satellite, the other satellites' trajectories can be considered the same as the solutions of the previous iteration, thus fixing their trajectories. In other words, only collisions with the nominal trajectories of these satellites need to be prevented, and the collision avoidance constraints are considered satisfied. Second, it is noted that most satellites are relatively far away during the orbit reconfiguration. Therefore, it is not necessary to impose collision avoidance constraints on any two satellites. To solve this problem, the second safe distance is defined, which is $d_{\text{safe}}, d_{\text{safe}} > d_{\min}$. In this way, only the neighbor satellites within the second safe distance at the previous iteration need to be identified. Then collision constraints are applied to these satellites. This strategy can greatly reduce the number of collision avoidance constraints and computation time. Based on the reference trajectory, the inter-satellite collision avoidance constraint can be modified, as shown in Eq. (33).

$$\boldsymbol{E}^{\mathrm{T}}\boldsymbol{G}\left(\boldsymbol{y}_p[k]-\bar{\boldsymbol{y}}_q[k]\right)-d_{\min}\parallel\boldsymbol{E}\parallel_2\geq 0,\quad k=0,\dots,K-1,$$
$$q\in\mathbb{N}_p,\quad p=2,\dots,M \tag{33}$$

where $\mathbb{N}_p$ is a set containing the neighbors of satellite p during the previous iteration.

$$\mathbb{N}_p=\{q|\exists k\in 0,\dots,K,\ \text{such that}\ \left\|\boldsymbol{G}\left(\bar{\boldsymbol{y}}_q[k]-\bar{\boldsymbol{y}}_p[k]\right)\right\|_2-d_{\text{safe}}\leq 0,$$
$$q<p\} \tag{34}$$

By decoupling the inter-satellite collision avoidance constraints, the decoupled convex optimization subproblem P5 is expressed by

Problem 5 *(decoupled convex problem).*

$$\begin{cases}\min_{\mathbf{u}_p,p=1,\dots,M}\sum_{p=1}^{M}\sum_{k=0}^{K-1}\parallel\boldsymbol{u}_p(k)\parallel_1\Delta t\\ \text{subject to}\\ \text{Eqs. (17), (19), (22), (24), (25), (33)}\\ \left\|\boldsymbol{y}_{p,i}[k]-\bar{\boldsymbol{y}}_{p,i}[k]\right\|_\infty\leq\delta_i,\ k=0,\dots,K-1,\ p=1,\dots,M\end{cases} \tag{35}$$

Similar to the coupled SCP approach, the L-1 penalty function is employed. Then the convex problem 6 (P6) can be written as

Problem 6 *(L-1 penalty-based decoupled convex problem).*

$$\begin{cases}\min_{\mathbf{u}_p,p=1,\dots,M}\sum_{p=1}^{M}\sum_{k=0}^{K-1}\parallel\boldsymbol{u}_p(k)\parallel_1\Delta t+\omega_{\text{eq}}\sum_{l=1}^{n_{\text{eq}}}\left|h_l'(\boldsymbol{y}_p)\right|\\ \quad+\omega_{\text{ineq}}\sum_{l=1}^{n_{\text{ineq}}}\left|g_l'(\boldsymbol{y}_p)\right|^{+}\\ \text{subject to}\\ \left\|\boldsymbol{y}_{p,i}[k]-\bar{\boldsymbol{y}}_{p,i}[k]\right\|_\infty\leq\delta_i,\ k=0,\dots,K-1,\ p=1,\dots,M\end{cases} \tag{36}$$

where equality constraints h_l' include Eqs. (17) and (22), inequality constraints g_l' include Eqs. (19), (24), (25) and (33).

5.2. Decoupled SCP -based satellite cluster reconfiguration

After decoupling the inter-satellite collision avoidance constraints, the decoupled convex optimization subproblem P6 based on the L-1 penalty function can be yielded. Combined with the framework of SCP, decoupled SCP -based satellite cluster reconfiguration can be obtained, as shown in Algorithm 2.

Similar to coupled SCP, the decoupled SCP algorithm also adopts a two-stage optimization strategy. Firstly, initializing parameters. However, note that unlike the coupled SCP method, the lowercase $\bar{\boldsymbol{y}}_p$ represents the initial nominal trajectory of the single pth satellite. $p=1,2,\dots M$.

In the first stage (line 1-8), collision avoidance constraints are not considered. A set of trajectories that approximately satisfy the satellite motion constraint are iteratively derived. It is important to point out that the states are not coupled between the satellites, so they can be solved in parallel to reduce computation time effectively. In line 4-7, each satellite independently solves the problem 6 without collision avoidance constraints and the computed trajectory is used as the nominal trajectory for the next iteration. When all satellites calculate their current trajectory, the trust domain (line 3) and nominal trajectory (line 6) are updated at the same time. Then, the iteration stops after Eqs. (17), (19) and (22) are satisfied. At the end of the first stage, the solution $\boldsymbol{y}_{p,i}$ is used as the nominal trajectory $\bar{\boldsymbol{y}}_p$ for the next stage.

In the second stage (lines10-26), the problem P6, including all constraints, is solved in parallel based on the initial nominal trajectory obtained in the first stage. As in Algorithm 1, the nominal trajectories used in each iteration are derived from the solutions obtained in the iteration.

Algorithm 2 Decoupled SCP-based Satellite Cluster Reconfiguration.

```
Initialization: $\bar{y}_p = y_{p,0}$, $\delta_i = \delta_0$, $i = 0$, $\eta$, $p = 1, 2, ..., M$
1  while $y_{p,i}$, $p = 1, 2, ...M$ does not satisfy Eqs. (17), (19), (22)
2     $i = i + 1$
3     $\delta_i = \eta \bullet \delta_{i-1}$
4     parfor $p = 1, 2, ..., M$
5        $y_{p,i} \leftarrow$ solution to P6 without Eqs. (24), (25), (33)
6        $\bar{y}_p = y_{p,i}$
7     end parfor
8  end while
9  $\delta_i = \delta_0$
10 unsuccessful set $= \{p|p = 1, \ldots, M\}$
11 while unsuccessful set $\neq \emptyset$
12    $i = i + 1$
13    $\delta_i = \eta \bullet \delta_{i-1}$
14    parfor $p \in$ unsuccessful set
15       $y_{p,i} \leftarrow$ solution to P6
16    end parfor
17    for $p = \{1, \ldots, M\}$ – unsuccessful set
18       $y_{p,i} = y_{p,i-1}$
19    end for
20    for $p = 1, 2, ..., M$
21       $\bar{y}_p = y_{p,i-1}$
22       if $y_{p,i}$ satisfy Eqs. (31), (32)
23          unsuccessful set = unsuccessful set – $\{p\}$
24       end if
25    end for
26 end while
27 return $y_{p,i}$, $p = 1, 2, ..., M$
```

It is worth noting that in Algorithm 2, the unsuccessful set (line10) needs to be defined, which refers to the satellites that do not satisfy the convergence conditions. For instance, if the p-th satellite meets the convergence conditions (line22), this satellite is considered to have found the final optimal trajectory. Then it is removed from the unsuccessful set (line 23). Now it is only necessary to iteratively compute the trajectories of the satellites in the unsuccessful set, while the others remain the same (Lines 17-19). Then, the algorithm does not stop until the unsuccessful set is empty (line 11). That is, all satellites have found the fuel-efficient trajectories that satisfy the convergence conditions and constraints at this moment (line 27).

It should be mentioned that the nominal trajectories are updated simultaneously after all satellites have completed the current iteration. Therefore, the trajectories of the satellites can be solved in parallel to decrease the running time dramatically. Unfortunately, Algorithm 2 may not converge, as the satellites are trying to avoid collision with each other. Therefore, all satellites need to be ranked so that satellites with low priority need to avoid satellites with high priority, while satellites with high priority do not. This strategy can effectively prevent the inter-satellite collision avoidance constraints from being over-conservative and raise the solution possibility.

Remark 4 *(computational complexity).* It makes sense to analyse the computational efficiency of two algorithms. The amount of resources required for executing a particular (computation or) algorithm is the computational complexity of that algorithm. In this paper, the computational complexity of Problem 4 and Problem 6 is related to the number of variables and constraints. The larger the number of variables and constraints included in the problem, the more resources are required for the computation.

Table 1
Initial parameters of reference orbit.

a/km	e	i/rad	Ω/rad	ω/rad	f/rad
6828.14	0.01	$\pi/3$	$\pi/6$	$\pi/4$	0

Table 2
Initial states of obstacles in LVLH coordinate system.

Spherical obstacles	Centre/m	Radius/m
Obstacle No.1	[-500,3500,0]	700
Obstacle No.2	[400,-5000,0]	900
Obstacle No.3	[0,-2000,0]	400
Obstacle No.4	[-2400,900,100]	600

For coupled SCP, the number of variables of the convex optimization problem is linearly related to the total number of satellites. However, the number of inter-satellite collision constraints increases quadratically with the number of satellites. Hence, the computational complexity is $O(n^2)$. In contrast, for decoupled SCP, the coupled convex problem (P4) is decomposed into M small independent problems. The number of convex subproblem variables is irrelevant to satellites' number, as are the dynamics constraints, control magnitude constraints and boundary conditions. However, the number of inter-satellite collision avoidance constraints increases to some extent as the number of satellites increases. There are two cases. The first case is that satellites can communicate globally, then the number of collision avoidance constraints is linearly related to the total number of satellites. Therefore, the computational complexity is $O(n)$. The second case is that satellites can only exchange information with nearby satellites. Then, the number of collision avoidance constraints is greatly reduced. The computational complexity is $O(n \bullet \chi)$, where χ represents the ratio of the number of satellites within the communication range to all satellites. Therefore, it can be seen that the decoupled SCP can effectively decrease the number of variables and constraints. This can reduce the computational complexity and running time.

6. Numerical simulations

In this section, comparative numerical simulations are performed to validate the effectiveness and efficiency of the coupled and decoupled SCP algorithms compared with the pseudo-spectral method. The optimal control problem of large-scale satellite cluster reconfiguration refers to finding the optimal trajectories that enable satellites to transfer from one set of orbits to another. At the same time, these trajectories need to achieve fuel efficiency and satisfy constraints such as inter-satellite collision avoidance and obstacle avoidance.

6.1. Simulation parameters

All numerical simulations are conducted using MATLAB2021a of Windows 10 Home Edition. The hardware environment is a personal computer with AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz and RAM 16.0 GB. The pseudo-spectral method is implemented based on the GPOPS [45] toolkit. The convex optimization subproblem is solved using the SeDuMi [46] solver and CVX [47].

The LEO orbit is chosen as the reference orbit for the satellite cluster reconfiguration, and its parameters are shown in Table 1. All satellites' initial and terminal positions are randomly generated, and their velocities can be calculated according to the J_2 invariant orbit conditions [48]. The total mission time is 5400 s. There are spherical obstacles on the way to the orbit transfer, and their states are shown in Table 2. Based on the above information, simulations are performed using the coupled SCP, the decoupled SCP, and the pseudo-spectral method. The basic parameters of the two SCP

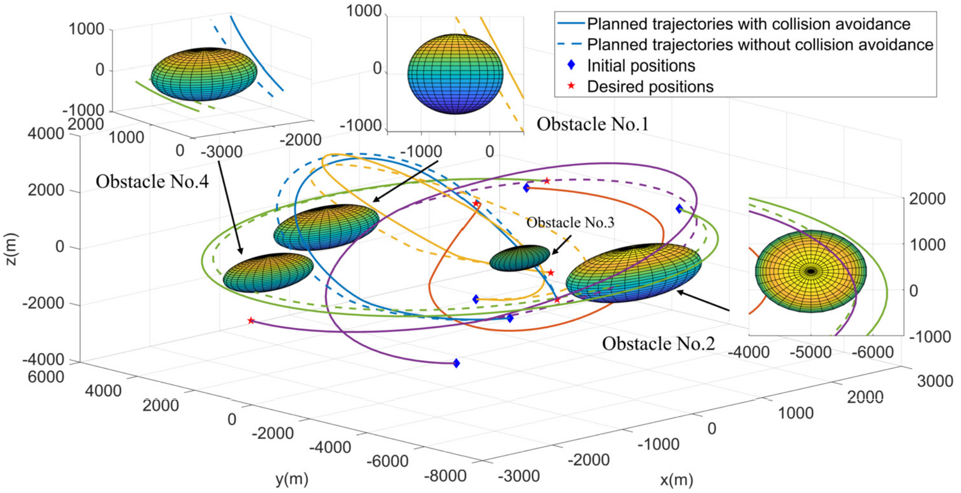


Fig. 6. Planned trajectories using coupled sequential convex programming method.

Table 3
Simulation parameters.

Parameter	Value
Transfer time t_f/s	5400
Time step Δt/s	60
Number of intervals K	90
Maximum control acceleration $T_{\max}/(\mathrm{m}\cdot\mathrm{s}^{-2})$	0.1
Minimum safe distance $d_{\min}$/m	50
Second safe distance d_{safe}/m	500
Initial trust region δ_0/m	5000
Trust region shrinkage factor η	0.2
Tolerance ε	10^{-3}

Table 4
Fuel consumption errors of clusters of 1-10 satellites using the coupled SCP and GPOPS methods.

Parameter	Value, m/s
Maximum fuel consumption error	0.0456
Average fuel consumption error	0.0326
Median fuel consumption error	0.0299
Standard deviation	0.0110

methods are shown in Table 3. $\omega_{\mathrm{eq}} = 100$, $\omega_{\mathrm{ineq}} = 9000$. For the GPOPS toolkit, the max iteration is 10. Minimum and maximum nodes per interval are 4 and 10, respectively. Other parameters of GPOPS are default.

6.2. Results and discussions

To demonstrate the effectiveness of the two SCP methods in fuel consumption, collision avoidance and run time, we compare the performance of the coupled SCP, decoupled SCP and GPOPS in the scenarios with 1-100 satellites.

To clearly show the trajectories, only the generated trajectories of 5 satellites are presented in Fig. 6 and Fig. 7. Fig. 6 and Fig. 7 illustrate the planned trajectories under the coupled and decoupled SCP algorithms. From these figures, it can be observed that the optimized trajectories of the two SCP methods are very similar, but their running time is different, which will be explained later. In both figures, the solid and dashed lines represent the trajectories with and without the obstacle avoidance, respectively. It can be seen that, when the obstacle avoidance constraints are not considered, several satellites collide with three obstacles (Obstacles No.1, No 2 and No 4). When the obstacle avoidance constraints are considered, the fuel-efficient trajectories (solid lines) obtained by the two SCP methods can both effectively avoid obstacles in the space. These results verify the effectiveness of the two SCP methods on both inter-satellite and obstacle collision avoidance.

Fig. 8 illustrates the variation of the minimum distances among 100 satellites in the reconfiguration under the coupled SCP and decoupled SCP. Fig. 9 shows the minimum distance among satellites and obstacles during the whole maneuver. The minimum safe distance is 50 m. During the whole reconfiguration, if the distance between the satellite and external obstacles and between satellites is less than 50 m, a collision is considered to have occurred. As can be seen in Fig. 8, the minimum distances among satellites using the coupled SCP and decoupled SCP methods are 61.617 m and 50.787 m during the whole reconfiguration. The average time-varying minimum distances are 332.756 m and 310.056 m. It can be concluded that all satellites satisfy the minimum safety distance constraint during reconfiguration. Hence, both SCP approaches can calculate the fuel-efficient trajectories that satisfy the collision avoidance constraints. As shown in Algorithm 1 and Algorithm 2, the iteration stops when the dynamics, control amplitude and boundary conditions are satisfied. This proves that the optimized trajectories theoretically satisfy the constraints in the optimization problem. When solving the optimal control problem of the large-scale satellite cluster reconfiguration, a set of collision-free, fuel-efficient trajectories and a set of control sequences are obtained. This set of control sequences is the control input in the simulation. The control sequences are applied to the original nonlinear dynamics model thus demonstrating that the control magnitude constraints and boundary condition constraints are satisfied. The effectiveness of the two methods in solving large-scale satellite cluster reconfiguration problems is demonstrated.

In order to validate the optimality and computational efficiency of the proposed algorithms, some simulations under GPOPS and two SCP methods are carried out in the cases of 1-100 satellites. Fig. 10 illustrates the average objective values of the satellite cluster reconfiguration corresponding to different numbers of satellites. In Fig. 10 (a), the number of satellites is from 1 to 10, while that is from 10 to 100 in Fig. 10 (b). From these figures, it can be concluded that there is no significant difference between the objective values obtained by SCP methods and GPOPS, and the deviation is less than 5% in all the scenarios.

As illustrated in Table 4 and Table 5, the maximum fuel consumption error of clusters of 1-10 satellites using the coupled SCP and GPOPS methods is 0.0456 m/s, while that for the decoupled SCP and GPOPS methods is 0.0533 m/s. The average fuel errors are 0.0299 m/s and 0.0379 m/s which means that the two SCP methods are comparable to GPOPS in terms of fuel consumption.

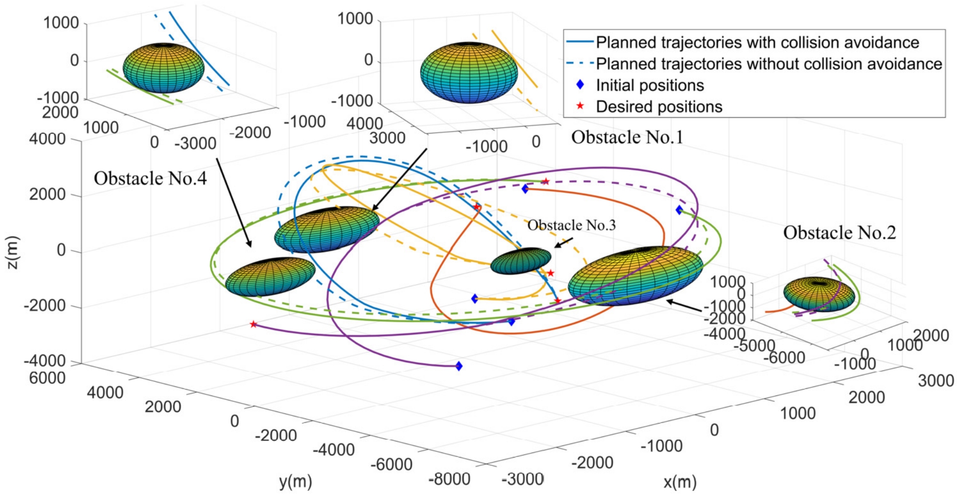
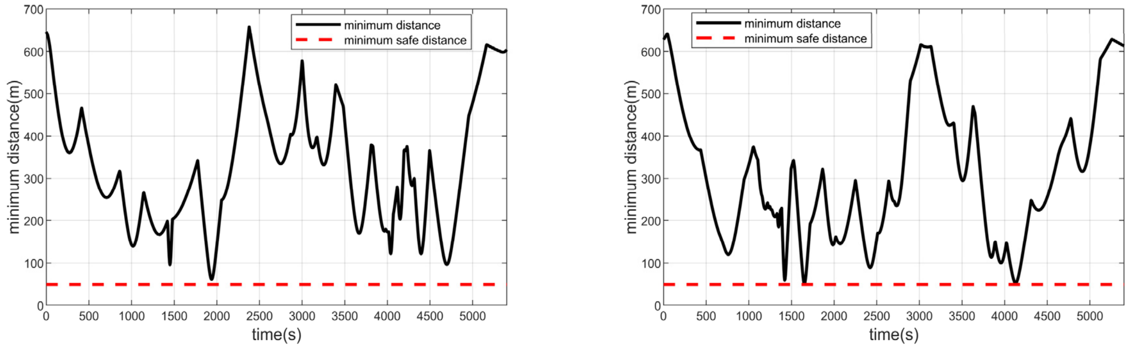


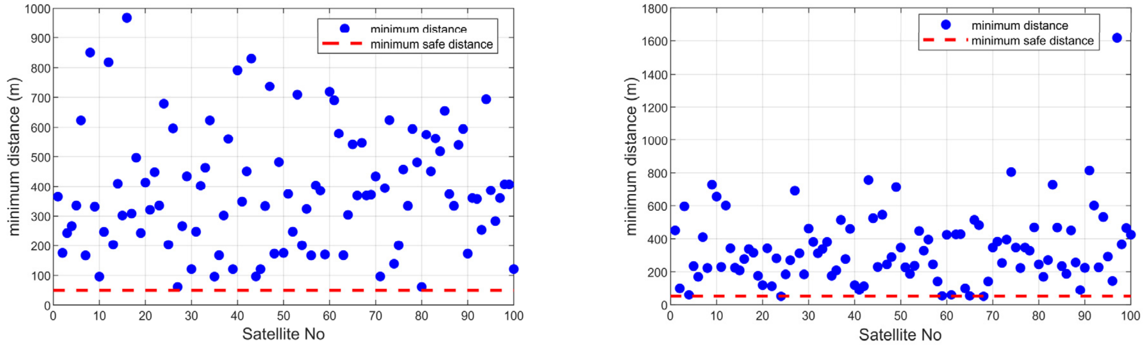
Fig. 7. Planned trajectories using decoupled sequential convex programming method.



(a) Minimum distance using coupled SCP

(b) Minimum distance using decoupled SCP

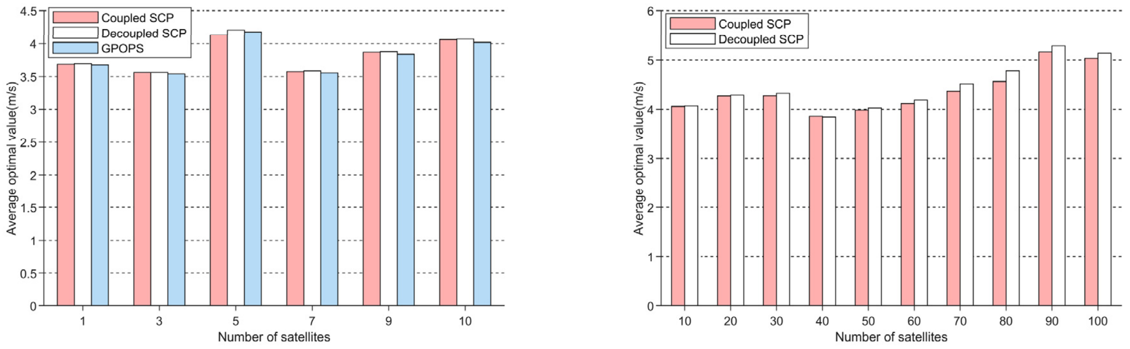
Fig. 8. Minimum distances among 100 satellites.



(a) Minimum distance using coupled SCP

(b) Minimum distance using decoupled SCP

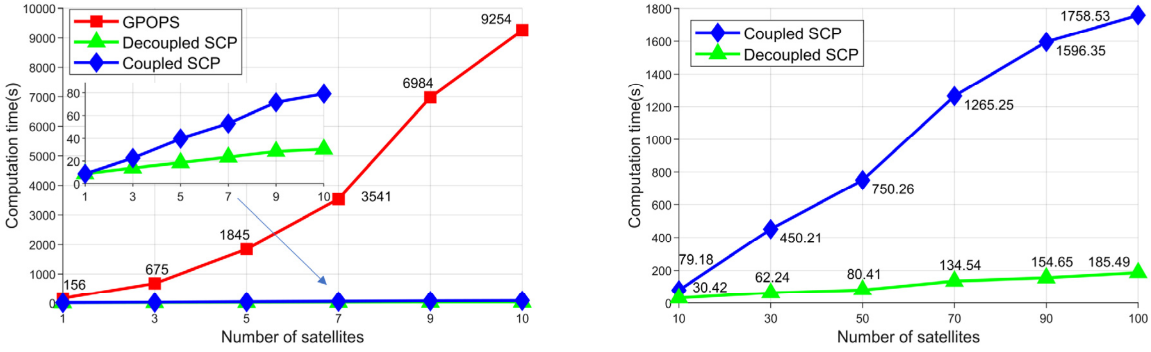
Fig. 9. Minimum distances among satellites and obstacles.



(a) Clusters of 1-10 satellites

(b) Clusters of 10-100 satellites

Fig. 10. Average objective values.



(a) Clusters of 1-10 satellites

(b) Clusters of 10-100 satellites

Fig. 11. Comparison of computation time.

Table 5
Fuel consumption errors of clusters of 1-10 satellites using the decoupled SCP and GPOPS methods.

Parameter	Value, m/s
Maximum fuel consumption error	0.0533
Average fuel consumption error	0.0379
Median fuel consumption error	0.0381
Standard deviation	0.0111

Table 6
Fuel consumption errors of clusters of 10-100 satellites using the coupled SCP and decoupled methods.

Parameter	Value, m/s
Maximum fuel consumption error	0.2165
Average fuel consumption error	0.1062
Median fuel consumption error	0.0990
Standard deviation	0.0638

The standard deviations are 0.0110 m/s and 0.0111 m/s, which indicates that the fuel consumption error distribution is relatively concentrated and the stability of the SCP method is excellent.

It is important to note that, GPOPS cannot find a feasible solution when the number of satellites is more than 10. As shown in Fig. 10(b) and Table 6, the maximum fuel consumption error of clusters of 10-100 satellites using the coupled SCP and decoupled methods is 0.2165 m/s and the average error is 0.1062 m/s. The coupled SCP saves about 3 percent fuel than the decoupled SCP. This proves that the decoupled SCP is almost as optimal as the coupled SCP. Therefore, it can be concluded that the optimality of the two SCP methods and GPOPS are similar in solving the optimal problem of large-scale satellite cluster reconfiguration trajectory planning.

Remark 5 *(simulation performance sensitivity).* The simulation performance is related to the number of intervals and the density of satellites. In the first case, the computational speed increases as the number of intervals decreases, but the accuracy of the solution drops. In the second case, if the satellite cluster is a tight cluster, the number of collision constraints will increase and the size of the problem will become larger. However, as mentioned before, the computational complexity of the decoupled SCP method is $O(n \cdot \chi)$, which means it is linear. Thus, the computational efficiency of the decoupled SCP method decreases slightly, but the variation is small.

Fig. 11 illustrates the computation time of the three methods. From Fig. 11 (a), it is observed that the run time of both SCP methods is much less than that of GPOPS and the gap becomes more significant as the number of satellites increases. It can also be seen that the run time required for GPOPS increases quadratically with the increasing number of satellites. For the scenario of 1 satellite, the GPOPS method takes 156 seconds, while the SCP method takes only 8.8 seconds. The computational efficiency is improved by 94%. At the same time, we can see that the GPOPS takes nearly 3 hours to solve the 10-satellite reconfiguration and it cannot be applied to the case with more than 10 satellites. In contrast, the computation time of the two SCP methods, whether coupled or decoupled, increases almost linearly with the increase of satellites. The required run time for solving the trajectory planning problem of 10 satellites is 79.18 s for the coupled SCP, while only 30.42 s for the decoupled SCP. This means that in this scenario, the proposed methods are almost 100 times faster than the GPOPS method. We now analyze the computational efficiency of the coupled SCP and decoupled SCP methods. As can be seen in Fig. 11 (b), for a 10-satellite scenario, the decoupled SCP requires half the computation time of the coupled SCP. Moreover, for 100 satellites, the coupled SCP takes 1758.53 s. Nevertheless, the decoupled SCP takes only 185.49 s, which is 10 times faster than the former.

As a result, it can be observed that the two SCP methods are more efficient than GPOPS. For the scenario with 10 satellites, the computational efficiency is improved by nearly 99% compared to the GPOPS method. At the same time, the decoupled SCP is the most efficient algorithm. For 100 satellites, the computational speed of the decoupled SCP is almost an order of magnitude higher than that of the coupled SCP. And it can always perform well in different numbers of satellites by decoupling the inter-satellite collision avoidance constraints and parallel computation.

In summary, by comparing the solutions of the above three methods, it can be summarized that these three methods obtain similar objective values in solving the optimal control problem of large-scale satellite cluster reconfiguration. That is, they are almost the same to optimality. In terms of computational efficiency, coupled and decoupled methods perform greatly better than GPOPS. As the number of satellites increases, 94 to 99 percent performance improvement over the pseudo-spectral method was achieved in terms of computational efficiency. In addition, the decoupled SCP is the most efficient and is not sensitive to the number of satellites. In the case of more than 50 satellites, the decoupled SCP method is almost 10 times faster than the coupled SCP in terms of calculation speed. Therefore, the decoupled SCP method has the advantage of solving the optimal problem of large-scale satellite cluster reconfiguration in real time.

7. Conclusion

In this paper, the trajectory planning algorithm based on sequential convex programming is developed for large-scale satellite cluster reconfiguration, which can quickly find the fuel-efficient trajectories under collision avoidance and thrust magnitude constraints. In this algorithm, the original nonconvex continuous optimal control problem containing nonconvex states and control constraints is described in a convex form by linearization and discretization. Meanwhile, the collision avoidance strategy between discrete points is proposed to ensure that trajectories are collision-free during the whole orbit transfer. Then, to solve the problem of infeasible solutions, the constraints are added to the objective function as penalty terms using the L1 penalty function. Further, the coupled and decoupled SCP methods are proposed for quickly obtaining fuel-efficient trajectories by solving a series of approximate convex subproblems. Comparative numerical simulations are performed for satellite clusters containing different numbers of satellites using coupled SCP, decoupled SCP and GPOPS. Simulation results show that the two SCP algorithms can guarantee collision avoidance during the reconfiguration and are almost identical to the GPOPS in fuel consumption. Compared with the GPOPS, the proposed methods can solve satellite cluster reconfiguration problems containing hundreds of satellites, where the decoupled SCP method achieves the best scalability and the highest computational efficiency. The proposed decoupled SCP method is promising in solving large-scale trajectory planning problems in real-time and onboard applications. The limitation of the proposed methods in this paper is that the transfer trajectories are planned in advance, which requires prior knowledge of the space environment during the whole reconfiguration. Satellites are unlikely to be able to avoid sudden external obstacles during the orbit transfer in time. Future work will investigate how to achieve real-time and enable emergency avoidance of surrounding obstacles by satellites in real time.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (62073102, 62203145), National Key R&D Program of China (2021YFC2202900), the China Postdoctoral Science Foundation (2022M710948).

Appendix A. Nomenclature

SCP	sequential convex programming
(X, Y, Z)	Earth-Centered-Inertial frame
(x, y, z)	coordinate values in the local-vertical/local-horizontal coordinate system
$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$	unit vectors of the local-vertical/local-horizontal coordinate system
$\boldsymbol{r}$	position vector of the leader
$\dot{\boldsymbol{r}}$	velocity vector of the leader
$\boldsymbol{r}_p$	position vector of the follower
$\dot{\boldsymbol{r}}_p$	velocity vector of the follower
$\boldsymbol{h}$	angular momentum vector per unit mass
r	geocentric distance magnitude of the leader
h	angular momentum magnitude of the leader
v_x	radial velocity of the leader
i	orbital inclination
Ω	right ascension of the ascending node
θ	argument of latitude
$\boldsymbol{\rho}_p = [x_p, y_p, z_p]^{\mathrm{T}}$	relative position vector in the local-vertical/local-horizontal coordinate system
$\dot{\boldsymbol{\rho}}_p = [\dot{x}_p, \dot{y}_p, \dot{z}_p]^{\mathrm{T}}$	relative velocity vector in the local-vertical/local-horizontal coordinate system
T	transpose of the vector
J_2	second harmonic coefficient of Earth
$\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^{\mathrm{T}}$	vector of rotation rates of the local-vertical/local-horizontal frame
$\boldsymbol{u}_p = [u_{px}, u_{py}, u_{pz}]^{\mathrm{T}}$	thrust acceleration on the follower satellite
k_{J2}	$\frac{3}{2} J_2 \mu R_e^2, 2.633 \times 10^{10}\ \mathrm{km}^2/\mathrm{s}^2$
μ	Earth's gravitational constant
R_e	Earth's equatorial radius
η	trust region shrinkage factor
δ_0	initial trust domain
δ_i	radius of the trust region at the ith iteration
χ	ratio of the number of satellites within the communication range to all satellites
ε	tolerance
$T_{\max}$	magnitude of the maximum thrust of each thruster
p	satellite p
t_{f}	mission time of the entire orbital transfer
M	number of satellites
$\boldsymbol{y}_p = (\boldsymbol{\rho}_p^{\mathrm{T}}, \dot{\boldsymbol{\rho}}_p^{\mathrm{T}})^{\mathrm{T}}$	state of the follower satellite p relative to the reference satellite
$\boldsymbol{o}$	state of the reference satellite
$d_{\min}$	minimum safe distance
O	set of states related to all obstacles
$\boldsymbol{O}_i$	state of the ith obstacle
$d_{\mathrm{obs},i}$	minimum allowable distance between satellites and the ith obstacle

$\boldsymbol{y}_{p,0}$	initial states
$\boldsymbol{y}_{p,\mathrm{f}}$	terminal states
K	intervals
t_0	start of the mission
t_f	end of the mission
Δt	time step
k	time step k
l_{qp}	line segment connects the centers of mass of any two satellites
l_p	line segment perpendicular to l_{qp}
$\bar{l}_{qp}$	dashed line connects the discrete points of the nominal trajectories of the two satellites
$\bar{\boldsymbol{y}}_p$	nominal trajectory
$\bar{\boldsymbol{Y}}_0$	nominal trajectory of the first iteration
$\bar{\boldsymbol{Y}}_i$	nominal trajectory of the ith iteration
$\boldsymbol{Y}$	states of all satellites at the discrete points
d_safe	second safe distance
ω_eq	penalty coefficients of equality constraints
ω_ineq	penalty coefficients of inequality constraints
$\mathbb{N}_p$	set containing the neighbors of satellite p

References

[1] C. Wei, X. Wu, B. Xiao, et al., Adaptive leader-following performance guaranteed formation control for multiple spacecraft with collision avoidance and connectivity assurance, Aerosp. Sci. Technol. 120 (2022) 107266.

[2] J. Wang, C. Zhang, J. Zhang, Analytical solution of satellite formation impulsive reconfiguration considering passive safety constraints, Aerosp. Sci. Technol. 119 (2021) 107108.

[3] W. Gao, K. Li, C. Wei, Satellite cluster formation reconfiguration based on the bifurcating potential field, Aerosp. 9 (3) (2022) 137.

[4] H. Zhang, P. Gurfil, Satellite cluster flight using on-off cyclic control, Acta Astronaut. (2015) 1–12, 106(jan.-feb.).

[5] F.Y. Hadaegh, S.J. Chung, H.M. Manohara, On development of 100-gram-class spacecraft for swarm applications, IEEE Syst. J. 10 (2) (2016) 673–684.

[6] A.M. Zou, A.H.J. de Ruiter, K. Dev Kumar, Distributed attitude synchronization control for a group of flexible spacecraft using only attitude measurements, Inf. Sci. 343 (344) (2016) 66–78.

[7] Y.-wen Zhang, L.-ping Yang, Y.-wei Zhu, et al., Modeling and analysis of dynamics for spacecraft relative motion actuated by inter-satellite non-contacting force, Aerosp. Sci. Technol. 43 (2015) 236–244.

[8] E. Lagona, S. Hilton, A. Afful, et al., Autonomous trajectory optimisation for intelligent satellite systems and space traffic management, Acta Astronaut. 194 (2022) 185–201.

[9] J.D. Seong, H.D. Kim, Optimization of collision avoidance maneuver planning for cluster satellites in space debris explosion situation, Proc. Inst. Mech. Eng., G J. Aerosp. Eng. 232 (3) (2018) 407–422.

[10] Z. Wang, Y. Xu, C. Jiang, et al., Self-organizing control for satellite clusters using artificial potential function in terms of relative orbital elements, Aerosp. Sci. Technol. 84 (2019) 799–811.

[11] S. Li, C. Liu, Z. Sun, Finite-time distributed hierarchical control for satellite cluster with collision avoidance, Aerosp. Sci. Technol. 114 (2021) 106750.

[12] E. Denenberg, P. Gurfil, Debris avoidance maneuvers for spacecraft in a cluster, J. Guid. Control Dyn. 40 (6) (2017) 1428–1440.

[13] H. Zhang, P. Gurfil, Cooperative orbital control of multiple satellites via consensus, IEEE Trans. Aerosp. Electron. Syst. (2018) 1.

[14] J. Luo, L. Zhou, B. Zhang, Consensus of satellite cluster flight using an energy-matching optimal control method, Adv. Space Res. 60 (9) (2017) 2047–2059.

[15] Y.K. Nakka, W. Hönig, C. Choi, et al., Information-based guidance and control architecture for multi-spacecraft on-orbit inspection, J. Guid. Control Dyn. 45 (7) (2022) 1184–1201.

[16] P. Lu, X. Liu, Autonomous trajectory planning for rendezvous and proximity operations by conic optimization, J. Guid. Control Dyn. 36 (2) (2013) 375–389.

[17] Y. Zhang, S. Tang, J. Guo, Two-stage cooperative guidance strategy using a prescribed-time optimal consensus method, Aerosp. Sci. Technol. 100 (2020) 105641.

[18] V. Arya, E. Taheri, J. Junkins, Electric thruster mode-pruning strategies for trajectory-propulsion co-optimization, Aerosp. Sci. Technol. 116 (2021) 106828.

[19] A. Sandberg, T. Sands, Autonomous trajectory generation algorithms for spacecraft slew maneuvers, Aerospace 9 (3) (2022) 135.

[20] K. Raigoza, T. Sands, Autonomous trajectory generation comparison for de-orbiting with multiple collision avoidance, Sensors 22 (18) (2022) 7066.

[21] E. Wilt, T. Sands, Microsatellite uncertainty control using deterministic artificial intelligence, Sensors 22 (22) (2022) 8723.

[22] N. Koeppen, I.M. Ross, L.C. Wilcox, et al., Fast mesh refinement in pseudospectral optimal control, J. Guid. Control Dyn. 42 (4) (2019) 711–722.

[23] A. Richards, T. Schouwenaars, J.P. How, et al., Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming, J. Guid. Control Dyn. 25 (4) (2002) 755–764.

[24] B. Hua, Y. Huang, Y. Wu, et al., Spacecraft formation reconfiguration trajectory planning with avoidance constraints using adaptive pigeon-inspired optimization, Sci. China Inf. Sci. 62 (7) (2019) 70209.

[25] S. Boyd, L. Vandenberghe, Convex Optimization, 2004.

[26] X. Yan, L. He, Unpowered approach and landing trajectory planning using second-order cone programming, Aerosp. Sci. Technol. 101 (2020) 105841.

[27] Y. Zhang, B. Zhu, M. Cheng, et al., Trajectory optimization for spacecraft autonomous rendezvous and docking with compound state-triggered constraints, Aerosp. Sci. Technol. 127 (2022) 107733.

[28] X. Liu, Convergence-guaranteed trajectory planning for a class of nonlinear systems with nonconvex state constraints, IEEE Trans. Aerosp. Electron. Syst. 58 (3) (2022) 2243–2256.

[29] S. Rahmanpour, R. Mahboobi Esfanjani, Energy-aware planning of motion and communication strategies for networked mobile robots, Inf. Sci. 497 (2019) 149–164.

[30] F. Scala, G. Gaias, C. Colombo, et al., Design of optimal low-thrust manoeuvres for remote sensing multi-satellite formation flying in low Earth orbit, Adv. Space Res. 68 (11) (2021) 4359–4378.

[31] Heekun Roh, Young-Jae Oh, Min-Jea Tahk, et al., L1 penalized sequential convex programming for fast trajectory optimization: with application to optimal missile guidance, Int. J. Aeronaut. Space Sci. 21 (2) (2020) 493–503.

[32] Y. Ma, B. Pan, C. Hao, et al., Improved sequential convex programming using modified Chebyshev–Picard iteration for ascent trajectory optimization, Aerosp. Sci. Technol. 120 (2022) 107234.

[33] X. Liu, S. Li, M. Xin, Mars entry trajectory planning with range discretization and successive convexification, J. Guid. Control Dyn. 45 (4) (2022) 755–763.

[34] Zhang Yingying, J. Huang, H. Cui, Trajectory design via convex optimization for six-degree-of-freedom asteroid powered landing, J. Guid. Control Dyn. 44 (4) (2021) 779–792.

[35] B. Benedikter, A. Zavoli, G. Colasurdo, et al., Convex approach to three-dimensional launch vehicle ascent trajectory optimization, J. Guid. Control Dyn. 44 (6) (2021) 1116–1131.

[36] Pei Pei, Jiang Wang, A new optimal guidance law with impact time and angle constraints based on sequential convex programming, Math. Probl. Eng. (2021) (2021) 1–15.

[37] Qiaoyang Xia, Shuang Liu, Mingyang Guo, et al., Multi-UAV trajectory planning using gradient-based sequence minimal optimization, Robot. Auton. Syst. 137 (2021) 103728.

[38] G. Xu, T. Long, Z. Wang, et al., Trust-region filtered sequential convex programming for multi-UAV trajectory planning and collision avoidance, ISA Trans. 128 (2022) 664–676.

[39] S. Sarno, J. Guo, M. D'Errico, et al., A guidance approach to satellite formation reconfiguration based on convex optimization and genetic algorithms, Adv. Space Res. 65 (8) (2020) 2003–2017.

[40] D. Morgan, S.J. Chung, F.Y. Hadaegh, Model predictive control of swarms of spacecraft using sequential convex programming, J. Guid. Control Dyn. 37 (6) (2014) 1725–1740.

[41] J. Chu, J. Guo, E. Gill, Decentralized autonomous planning of cluster reconfiguration for fractionated spacecraft, Acta Astronaut. 123 (2016) 397–408.

[42] J.A. Kechichian, Motion in general elliptic orbit with respect to a dragging and precessing coordinate frame, J. Astronaut. Sci. 46 (1) (1998) 25–45.

[43] G. Xu, D. Wang, Nonlinear dynamic equations of satellite relative motion around an oblate Earth, J. Guid. Control Dyn. 31 (5) (2008) 1521–1524.

[44] T. Antczak, The exact l_1 penalty function method for constrained nonsmooth invex optimization problems, in: D. Hömberg, F. Tröltzsch (Eds.), System Modeling and Optimization, vol. 391, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 461–470.

[45] M.A. Patterson, A.V. Rao, GPOPS-II: a MATLAB software for solving multiple-phase optimal control problems using hp-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming, ACM Trans. Math. Softw. 41 (1) (2010) 1–37.

[46] J.F. Sturm, Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones (updated for version 1.05), Optim. Methods Softw. (1999) 11.

[47] M. Grant, S. Boyd, CVX: MATLAB software for disciplined convex programming, http://cvxr.com/cvx, 2014.

[48] D. Morgan, S.J. Chung, L. Blackmore, et al., Swarm-keeping strategies for spacecraft under J2 and atmospheric drag perturbations, J. Guid. Control Dyn. 35 (5) (2012) 1492–1506.