Aerospace Science and Technology 93 (2019) 105295

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Short communication

Fast preliminary design of low-thrust trajectories for multi-asteroid exploration

Zichen Fan [a], Mingying Huo [a,*], Naiming Qi [a], Ye Xu [a], Zhiguo Song [b]

[a] School of Astronautics, Harbin Institute of Technology, Harbin 150001, China
[b] China Academy of Launch Vehicle Technology (CALT), Beijing 100076, China

ARTICLE INFO

Article history:
Received 1 February 2019
Received in revised form 21 May 2019
Accepted 11 July 2019
Available online 16 July 2019

Keywords:
Low-thrust trajectory design
Asteroid sequence selection
Multi-asteroid exploration
Finite Fourier series
Monte-Carlo tree search

ABSTRACT

Multiple-asteroid exploration with low-thrust propulsion requires the design of transfer trajectory and the selection of the visiting sequence, as well as the estimation of the propellant budget. To this end, this paper presents a method to generate transfer trajectories and visiting sequences rapidly using finite Fourier series (FFS) and Monte-Carlo Tree Search (MCTS). The FFS method can generate the transfer trajectory rapidly, which can provide suitable initial approximations, leading to more accurate trajectory optimizations. This study adopts the MCTS algorithm for asteroid sequence selection in the multi-asteroid exploration. By comparing with the traversal algorithm, the greedy algorithm and the tree search algorithm with the trimming strategy, the numerical results show that the MCTS can be used to obtain a quasi-optimal sequence with higher probability and less computation time. Consequently, a method combining FFS and MCTS can rapidly acquire the quasi-optimal visiting sequences with a large probability and the suitable initial trajectory for a multi-asteroid exploration mission. This is very important for the rapid feasibility assessment of hundreds of flight scenarios at the preliminary mission design stage.

1. Introduction

Spacecraft with low-thrust propulsion systems have been used in several deep space exploration missions, such as Deep Space 1, Hayabusa, DAWN etc. The trajectory of a spacecraft plays a key role in the flight mission and scenario design [1]. In a low-thrust trajectory, ion engines need to function for most of the mission time. Therefore, propellant consumption is the most important performance characteristic to be considered in the mission trajectory design. Designers need to optimize continuous thrust profiles, which makes the analysis of low-thrust missions mathematically and computationally challenging [2]. Common techniques for low-thrust trajectory design [3] require a few initial approximations, but the generation of a suitable initially approximated trajectory during the preliminary mission design phase is not trivial [2].

Whether it is to gain an early understanding of the potential threat from asteroids or to explore asteroids for resources, the multi-asteroid exploration is a popular and rapidly progressing field that can enable these endeavors. The application of low-thrust spacecraft in multi-asteroid exploration has many advantages. In a mission scenario, each asteroid has a corresponding optimal transfer trajectory because each asteroid has different orbital parameters [4]. However, when selecting several asteroids from a large group of asteroids for exploration, a significant amount of time and calculation is needed to obtain the optimal visit sequence and generate the visit trajectory because there are several optional sequences and the trajectory generation is complex. Therefore, it is of great significance that multi-asteroid exploration can quickly select the exploration sequence and generate the trajectories to provide an initial solution for subsequent precise trajectory optimizations.

The mission analysis phase is lengthy and difficult, and is usually tackled by identifying the space trajectory that minimizes a scalar performance index [5]. In addition, the dynamic constraints of the spacecraft need to be considered in the trajectory optimization problem [6]. To make the process of obtaining a suitable initial approximation for low-thrust trajectory design more efficient, several researchers have begun using an inverse approach to satisfy the equations of motion (EoMs) and boundary conditions (BCs). In recent years, several studies have proposed shape-based (SB) methods for rapid trajectory generation. In these methods, the trajectory shapes are presupposed as some specific function forms, and the unknown parameters of the assumed fixed functions are calculated using the BCs. Petropoulos and Longuski [7] first proposed the SB method and utilized exponential sinusoids to model low-thrust trajectories to obtain the corresponding thrust profiles. Furthermore, Pascale and Vasile [8] used a set of parameterized

* Corresponding author.
E-mail address: huomingying123@gmail.com (M. Huo).

https://doi.org/10.1016/j.ast.2019.07.028
1270-9638/

pseudo-equinoctial elements to produce the preliminary design of low-thrust multiple gravity-assist trajectories. Wall and Conway [9] used an inverse polynomial to model the radius of a planar trajectory in polar coordinates, and the methods satisfied all the considered BCs. Wall applied the method to cylindrical coordinates [10]. Xie et al. [11] provided a shaping approximation for low-thrust trajectories, which considered the trajectory constraints between two coplanar elliptical orbits. Moreover, Novak and Vasile [2] presented a novel shaping method, based on a three-dimensional description of the trajectory in spherical coordinates. Gondelach and Noomen [12] presented a hodographic-shaping method that considered velocity hodographs for the low-thrust transfer problem. Xie et al. [13] provided a new combination of the elevation angle and radius shapes for three-dimensional low-thrust trajectories, valid for large out-of-plane motion. Zeng et al. [14] proposed an analytic shape-based approach for trajectory design. Peloni et al. [15] presented a combination of exponential and sinusoidal terms to approximate the trajectories for near-earth asteroids in rendezvous missions. Taheri and Abdelkhalik presented a method for the rapid generation of two-dimensional low-thrust trajectories that utilizes FFS [16] and used the FFS approximation on two-dimensional three-body dynamic models [17]. Further, Taheri and Abdelkhalik extended the FFS approximation to three dimensions [18]. More recently, Taheri et al. [19] have used the FFS approximation to represent trajectories in spherical coordinates and enforce thrust constraints. These SB methods provide new ideas for the rapid generation of trajectories.

For the selection of the exploration sequence for multi-asteroid exploration, many research groups have proposed methods to solve problems associated with the exploration sequence, flight path optimization, and propellant budget. Ulybyshev [20] adopted the inner-point algorithm to optimize continuous thrust orbit transfers. Betts [21] used the sequential quadratic programming method to study orbit transfer optimization, which reflects the advantages and features of having low thrust. Olds et al. [22] reported that differential evolution (DE) is a global method suitable for trajectory optimization. However, the choice of routine tuning parameters influences the performance of the method. Hull [23] converted the optimal control problem into a parameter optimization problem, replacing the control and state history with control and state parameters and forming a history by interpolation. Zhu et al. [24] proposed a hybrid algorithm of particle-swarm optimization and DE to select candidate asteroids and achieve a solution to the energy-optimal path mission. In recent years, several new learning methods have been proposed, including adaptive iterative learning [25], the neural network algorithm etc. Among the neural network algorithms, the MCTS algorithm, which has been widely used in the field of artificial intelligence, shows outstanding advantages. Through its ingenious search method, combined with advanced artificial intelligence algorithms, it can solve complex problems at high computational speeds. Silver et al. proposed a method to use MCTS in Alpha Go [26], chess, and "shogi" [27]. Inspired by [26] and [27], in this study, MCTS is applied towards selecting the exploration sequence for multi-asteroid exploration.

This work takes advantage of the FFS method to generate the flight trajectory at a very short computational time and the MCTS algorithm to determine the exploration sequence in multi-asteroid exploration. Unlike the FFS method with fixed time [18], this work extends the FFS method to the unconstrained time case. The feasibility of the trajectory obtained by the FFS method can be proved by integration. Generally, in multi-asteroid exploration, the optimal exploration sequence can be computed by the traversal algorithm. However, the increasing number of visited asteroids will only increase the computation time. In such a scenario, FFS and MCTS will prove advantageous as they offer high computational speeds.

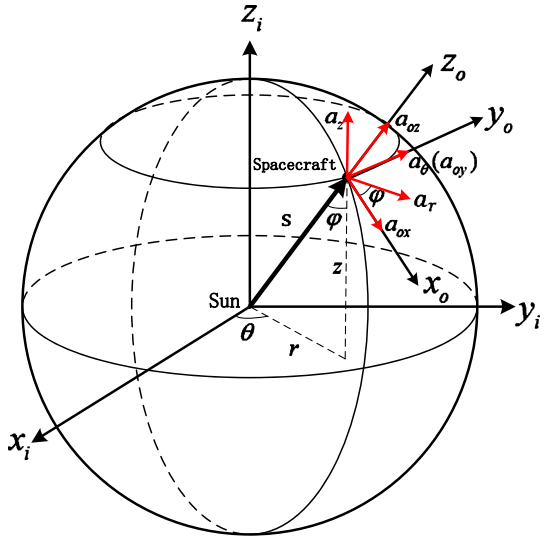

Fig. 1. Cartesian and cylindrical coordinate systems.

This paper is organized as follows. In Section 2, the problem description is presented, and the coordinate systems (CSs), EoMs, and BCs are explained. In Section 3, the FFS method with free time is briefly described. The MCTS algorithm for asteroid sequence selection is presented in Section 4. Finally, Section 5 compares the solutions obtained by four different algorithms through numerical simulations.

2. Problem description

The propellant consumption is used as the performance index in this work. This section presents the orbital dynamics and constraints of orbital transformation and subsequently provides the additional constraints for multi-asteroid exploration.

The thrust model and orbit dynamics of the spacecraft are described in two Cartesian coordinate systems, which are the orbital reference frame $o_o x_o y_o z_o$ and the heliocentric ecliptic inertial frame $o_i x_i y_i z_i$. As illustrated in Fig. 1, the origin o_o of the orbital frame is located at the center of the mass of the spacecraft. The z_o-axis is aligned with the sun–spacecraft direction. The y_o-axis is perpendicular to the z_o-axis and the normal of the ecliptic plane, and the x_o-axis forms a right-handed triad. The origin o_i of the heliocentric ecliptic inertial frame is located at the center of mass of the sun, and the x_i-axis is defined in the direction of the sun equinox. The z_i-axis is aligned with the normal of the ecliptic plane and is directed towards the north ecliptic pole. The y_i-axis forms a right-handed system. In addition to these Cartesian coordinate systems, cylindrical coordinates (r, θ, z) are used to describe the position of the spacecraft in the FFS method. As shown in Fig. 1, r is the radial distance, θ is the azimuth angle, and z is the altitude of the spacecraft.

The EoMs in the reference coordinates [18][28] are given as

$$\begin{aligned} &\ddot{r} - r\dot{\theta} + \mu_\odot r/s^3 = a_r \\ &r\ddot{\theta} + 2\dot{r}\dot{\theta} = a_\theta \\ &\ddot{z} + \mu_\odot z/s^3 = a_z \end{aligned} \tag{1}$$

where $s = \sqrt{r^2 + z^2}$ is the distance between the centers of the spacecraft and the sun, $\mu_\odot$ is the gravitational parameter of the central body (the sun), and a_r, a_θ and a_z are the components of propulsive acceleration. Therefore, the required propulsive acceleration in the EoMs, a, can be computed as

$$a = \sqrt{a_r^2 + a_\theta^2 + a_z^2} \tag{2}$$

The performance index of trajectory optimization is the total ΔV, and it can be computed as

$$\Delta V = \int_0^T a\, dt \tag{3}$$

where T is the total flight time in each transfer.

For problems involving the orbital transfer of two asteroids, the following 12 BCs are applicable to the launch window:

$$\begin{aligned} &r(\tau=0)=r_i, r(\tau=1)=r_f, r'(\tau=0)=T\dot{r}_i, r'(\tau=1)=T\dot{r}_f \\ &\theta(\tau=0)=\theta_i, \theta(\tau=1)=\theta_f, \theta'(\tau=0)=T\dot{\theta}_i, \theta'(\tau=1)=T\dot{\theta}_f \\ &z(\tau=0)=z_i, z(\tau=1)=z_f, z'(\tau=0)=T\dot{z}_i, z'(\tau=1)=T\dot{z}_f \end{aligned} \tag{4}$$

where subscripts "i" and "f" denote the initial and final conditions, respectively, t is the flight time, and $\tau = t/T \in [0,1]$ is the dimensionless time. The symbol $\dot{}$ denotes a derivative with respect to time, t, and the superscript $'$ represents a derivative with respect to dimensionless time, τ.

In multi-asteroid exploration, the BCs should be satisfied for each orbital transfer because the final conditions of a previous orbital transfer are the initial conditions in the subsequent orbital transfer. The launch window for the subsequent orbital transfer also needs to be determined by the initial launch window and the time taken for the previous orbital transfer, as follows:

$$t_{l(n)} = t_{li} + T_1 + T_2 + \cdots + T_{n-1}, n \geq 2 \tag{5}$$

where t_{li} is the initial launching window, $t_{l(n)}$ is the launch window for the nth orbital transfer, and $T_1, T_2, \cdots, T_{n-1}$ are the total flight time in each transfer. When $n = 1$, $t_{l(1)} = t_{li}$. Equation (5) is used to compute the launch window in each transfer when the spacecraft visits several asteroids simultaneously and makes a rendezvous with the asteroids only for a short time. If the spacecraft makes a rendezvous with the asteroid for a specified time, the residence time needs to be added to Eq. (5). Therefore, in the problem of multi-asteroid exploration, the trajectory generated for the previous trajectory will have a high impact on that generated for the subsequent trajectory, which should be considered in the overall trajectory generation and asteroid sequence selection.

3. Finite Fourier series method with free time

3.1. Fourier approximation

According to Taheri and Abdelkhalik's article [18], radius r in cylindrical coordinates is approximated with FFS as follows:

$$r(\tau) = \frac{a_0}{2} + \sum_{n=1}^{n_r} \{a_n \cos(n\pi\tau) + b_n \sin(n\pi\tau)\} \tag{6}$$

where n_r is the number of Fourier terms for each coordinate, and a_n and b_n are the unknown Fourier coefficients. The other two coordinates, θ and z, can be processed similarly (see [18] for details).

Taheri and Abdelkhalik [18] expressed the first four coefficients of each Fourier approximation in terms of the BCs to satisfy them naturally. The number of Fourier coefficients is $2(n_r + n_\theta + n_z) + 3$, constituted by the a_n, b_n, and a_0 terms present in Eq. (6) and in similar equations related to θ and z. The twelve BCs allow the number of unknown Fourier coefficients to be reduced to $2(n_r + n_\theta + n_z) - 9$. Then, the Fourier approximation of coordinate r is written as follows (the other two coordinates θ, z can be processed similarly):

$$r(\tau) = F_r + C_{a0}a_0 + \sum_{n=3}^{n_r} \{C_{an}a_n + C_{bn}b_n\} \tag{7}$$

where

$$\begin{aligned} F_r =& \frac{1}{2}(r_i - r_f)\cos(\pi\tau) + \frac{T}{2\pi}(\dot{r}_i - \dot{r}_f)\sin(\pi\tau) \\ &+ \frac{1}{2}(r_i + r_f)\cos(2\pi\tau) + \frac{T}{4\pi}(\dot{r}_i + \dot{r}_f)\sin(2\pi\tau) \\ C_{a0} =& \frac{1}{2}(1 - \cos(2\pi\tau)) \\ C_{an} =& \begin{cases} \cos(n\pi\tau) - \cos(\pi\tau); & \text{when } n \text{ is odd} \\ \cos(n\pi\tau) - \cos(2\pi\tau); & \text{when } n \text{ is even} \end{cases} \\ C_{bn} =& \begin{cases} \sin(n\pi\tau) - n\sin(\pi\tau); & \text{when } n \text{ is odd} \\ \sin(n\pi\tau) - 0.5n\sin(2\pi\tau); & \text{when } n \text{ is even} \end{cases} \end{aligned}$$

The fulfillment of the BCs can be verified by substituting $\tau = 0$ and $\tau = 1$ into Eq. (7). Then, the first- and second-order τ-derivatives of Fourier approximation of the coordinates can be obtained (see [18] for details).

Evaluation at arbitrary discretization points is required to provide a sufficient number of constraints of propulsive acceleration along the trajectory. The Legendre–Gauss distribution of the discretization points, which are the root of the mth-degree Legendre polynomial, is adopted,

$$\tau_1 = 0 < \tau_2 < \cdots < \tau_{m-1} < \tau_m = 1 \tag{8}$$

Because the dimensionless-time vector is represented as a column vector, coordinates (r, θ, z) and their associated first-order and second-order τ-derivatives can be written in a compact matrix notation form. Coordinate r is again taken as an example,

$$[r]_{m\times 1} = [A_r]_{m\times(2n_r-3)}[X_r]_{(2n_r-3)\times 1} + [F_r]_{m\times 1} \tag{9}$$

where $[F_r]$, $[X_r]$ and $[A_r]$ are detailed in Ref. [18].

Using the matrix notation form of the coordinates (r, θ, z) and their associated first and second τ-derivatives, the equations of the components of propulsive acceleration can be written in a compact matrix notation form,

$$\begin{aligned} [a_r]_{m\times 1} &= a_r\left(X_r, [r], [r'], [\theta'], [z]\right) \\ [a_\theta]_{m\times 1} &= a_\theta\left(X_\theta, [r], [r'], [\theta'], [\theta'']\right) \\ [a_z]_{m\times 1} &= a_z\left(X_z, [r], [z], [z'']\right) \end{aligned} \tag{10}$$

and Eq. (2) can be written as

$$[a] = \sqrt{[a_r]^2 + [a_\theta]^2 + [a_z]^2} \leq a_{\max} \tag{11}$$

where X_r, X_θ and X_z are the unknown Fourier coefficients of each coordinate, and a_{max} is the maximum limit of the propulsive acceleration value. For the time-fixed rendezvous problem, the number of variables is $2(n_r + n_\theta + n_z) - 9$.

In this study, the goal is to visit multiple asteroids, which is a time-free rendezvous problem. Thus, the nonlinear programming (NLP) problem should be written as

$$\begin{aligned} &\min_{[X_r],[X_\theta],[X_z],[T]} \quad \Delta V \\ &\text{s.t.} \quad [a] \leq a_{\max} \end{aligned} \tag{12}$$

where X_r, X_θ and X_z are the unknown Fourier coefficients of each coordinate, a_{max} is the maximum limit of the propulsive acceleration value, T is the total flight time, and $2(n_r + n_\theta + n_z) - 8$ is the number of variables.

3.2. Unknown coefficient initializations

This section describes the techniques for initializing variables that need to be optimized. Ref. [18] used the Cubic Polynomial (CP) to initialize the unknown Fourier coefficients. The CP is used to generate approximations of the coordinates (r, θ, z) at m Legendre-Gauss discretization points. And than fit the considered Fourier series functions to this set of discrete points to calculate the unknown Fourier coefficients. An initial approximation for the unknown Fourier series parameters is obtained from

$$[X_r]_{(2n_r-3)\times 1} = \left([A_r]_{n\mathrm{App}\times(2n_r-3)}\right)^{-1}\left([r_{\mathrm{App}}] - [F_r]\right) \tag{13}$$

where r_{App} is the vector of the discretized approximation and n_{App} is the number of discretized data points that is different from $m(n_{\mathrm{App}} > m)$. A CP is used for the approximation of r_{App}, as follows:

$$r_{\mathrm{App}}(\tau) = a + b\tau + c\tau^2 + d\tau^3 \tag{14}$$

where a, b, c, d are the polynomial coefficients of each coordinate, which can be calculated by the BCs in Eq. (6). When the Legendre–Gauss distribution of points is considered, the dimensionless time vector becomes

$$\tau_{\mathrm{App},0} = 0 < \tau_{\mathrm{App},1} < \cdots < \tau_{\mathrm{App},(n_{\mathrm{App}}-1)} = 1 \tag{15}$$

$[r_{\mathrm{App}}]$ can be acquired by substituting $\tau = \tau_{\mathrm{App}}$ into Eq. (14). The other two coordinates, θ and z, can be processed similarly. This is different from [18] where the initial unknown Fourier coefficients of z are set to zero in all cases.

In this study, the total flight time T is free and needs to be optimized as the other coefficients. The approximated flight time T_{APP} of each orbit transfer gives an inaccurate initial estimate, and it is estimated under the following assumptions: (1) The orbits of two consecutive transfer asteroids are coplanar. (2) The orbits of two consecutive transfer asteroids are circular with a radius of the semi-major axis. (3) The orbit follows a Hohmann trajectory transfer. The ΔV of the orbit transfer is calculated under these assumptions, and then divided by a_{max} to produce an initial estimate of the flight time. Because the above assumptions produce an estimate of the time period of the orbital transfer far less than the actual time, the result of the above calculation of flight time is multiplied by 2,

$$T_{\mathrm{APP}} = \frac{\left|\sqrt{\frac{\mu_\odot}{r_1}\frac{2r_2}{r_1+r_2}} - \sqrt{\frac{\mu_\odot}{r_1}\frac{2r_1}{r_1+r_2}}\right|}{a_{\max}} \times 2 \tag{16}$$

where r_1 and r_2 are the semi-major axis of the initial and final orbits, respectively, and a_{max} is the maximum limit of propulsive acceleration.

4. Monte-Carlo tree search algorithm

MCTS combines the generality of a stochastic simulation and the accuracy of a tree search procedure. This has been shown to work well in Alpha Go [26], chess, and shogi (Japanese chess) [27]. The optimized MCTS algorithm is more complex, but it solves optimization problems with very large search depths and widths. On the other hand, the basic MCTS algorithm is relatively simple and can solve optimization problems with large search depths and widths at high computational speeds. In this study, the basic MCTS algorithm is presented to select the quasi-optimal visiting sequences in multi-asteroid exploration. The MCTS is applicable to both deep space exploration missions that return to Earth after exploration and that do not return to Earth after exploration.

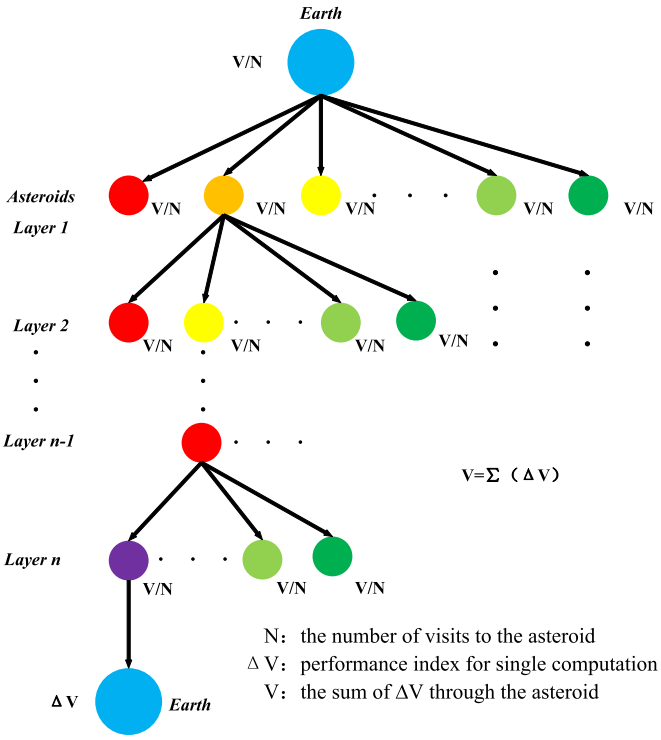

Fig. 2. Search tree in MCTS for multi-asteroid exploration.

Therefore, to enhance the complexity of the mission, this study introduces the mission of returning to the earth after exploration.

The main concept of MCTS is the search, and the rule of the search is the upper confidence bound (UCB). The main formula when MCTS combined with the UCB (upper confidence bound applied to trees (UCT)) is as follows:

$$UCB(S_i) = -\Delta\overline{V_i} + c\sqrt{\frac{\ln N}{n_i}} \tag{17}$$

where S_i represents the node in the search tree, $\overline{V_i}$ represents the average ΔV through the node, c is an adjustable parameter, N is the total number of iterations, and n_i is the number of iterations passing through the node. The search rule of UCT is to select the node with the maximum UCB value; however, when selecting the visiting sequences for deep-space exploration, the least propellant-consuming sequence should be selected. Therefore, a negative sign is added to $\Delta\overline{V_i}$.

The search tree in MCTS for multi-asteroid exploration is shown in Fig. 2. Earth is the root node, and each of the rest of the nodes represents an asteroid. The first layer consists of all the selected asteroids and the subsequent layer consists of the remaining asteroids after removing all the previously visited asteroids. If n asteroids are selected for exploration, there are n layers, and finally, the spacecraft returns to Earth. V in V/N represents the sum of ΔV through this asteroid and N in V/N is the number of visits to the asteroid. In the initial case, N and V are both 0. The entire search process is divided into four steps: selection, expansion, random simulation, and back propagation.

The selection phase, as shown in Fig. 3, starts from the root node — Earth. The selection of the nodes in each layer involves two cases. In case one, there are several nodes that have not been visited in that layer; i.e., there are several nodes with $N = 0$. In case two, there is no node that has not been visited in that layer; i.e., the N of all nodes in that layer is not zero.

In case one, the selection process involves selecting one of these unvisited nodes in order. After the selection of this layer is com-

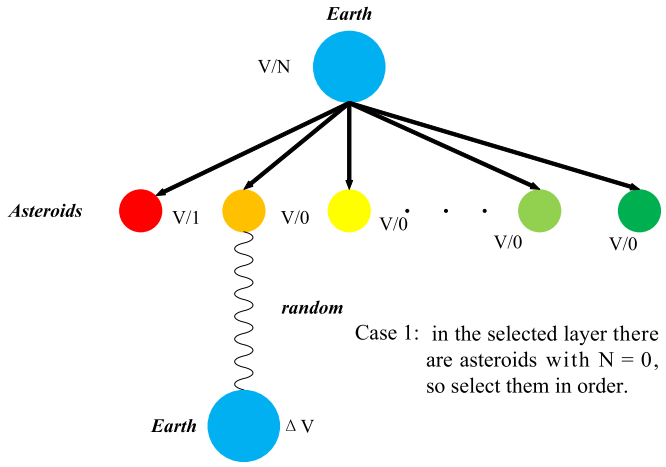

Fig. 3. Selection and random simulation.

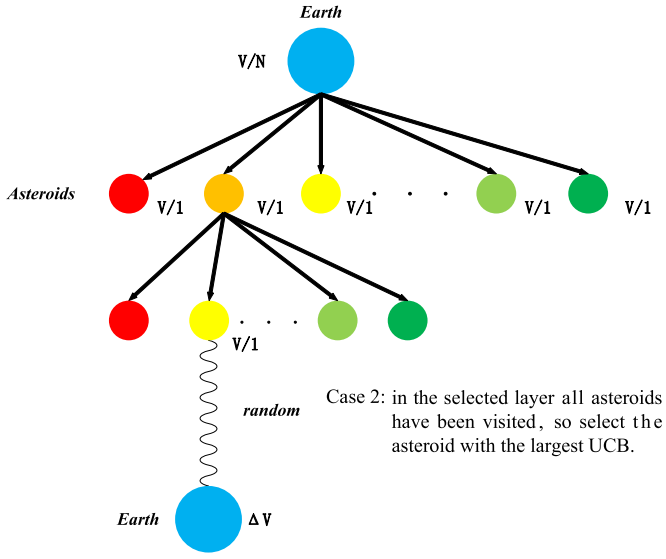

Fig. 4. Expansion and random simulation.

pleted, the random simulation process begins. First, the transfer trajectory from Earth to the target asteroid is generated. Then, $n-1$ asteroids are randomly selected from the remaining asteroids, and the full exploration trajectory is generated. Finally, the spacecraft returns to Earth. The value of ΔV for the entire orbital transfer process is obtained.

In case two, the UCB values of all nodes in that layer are calculated, and the node with the largest UCB is selected to visit. When the selection is completed, the expansion process begins as shown in Fig. 4. The selected node continues to expand to the next level, and node selection and random simulation are executed. The principle of node selection is the same as above.

If the asteroid cannot meet the constraints to achieve orbital transfer under the current conditions, then 1 is assigned to its N, and ∞ is assigned to its V. In this way, it can be seen from Eq. (17) that the UCB value of the node will be negative infinite, which indicates that the node has been visited and will not be selected in the future. Furthermore, one node is selected from the remaining nodes in accordance with the order or UCB to visit.

When the random simulation of a node is completed, back propagation process is executed. The resulting ΔV is replicated on all the clearly selected nodes in the current search tree, and the values of N of all the clearly selected nodes are increased by one,

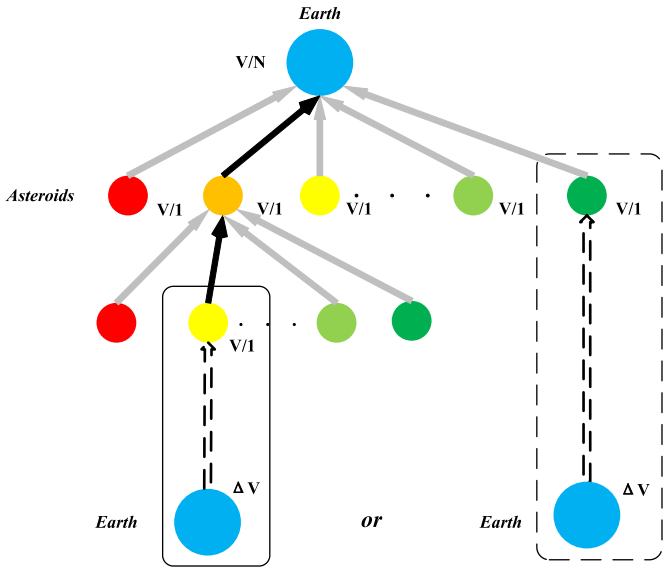

Fig. 5. Back propagation.

as shown in Fig. 5. For those nodes selected in the random simulation process, the values of V and N are not updated.

When the N layer is expanded, the whole visiting sequence selection is completed. Each time the back propagation is completed, an iterative process is completed. The MCTS program is terminated by controlling the number of iterations. However, to ensure that the complete visiting sequence is generated through computation, when the number of iterations without the random simulation process is recorded, all the nodes of this iteration are clearly selected, i.e., no sequences are chosen at random. When the number of iterations reaches a given value, the computation is complete.

To ensure that the visiting sequence with the minimum propellant consumption is identified during the entire search process, when the MCTS is completed, the selection of asteroids is started in the layer N. The V/N value of all the visited asteroids in the N layer is calculated, from which the asteroid with the lowest V/N value is selected. The asteroid from the bottom of the search tree upward is selected to determine the sequence with minimum propellant consumption.

5. Numerical simulations

In order to verify the effectiveness of the proposed method, the problem of the third Global Trajectory Optimization Competition (GTOC-3) [29] was selected as a simulation example. The GTOC-3 proposed that a spacecraft launched from Earth must first rendezvous with three asteroids from a specified group of near-Earth asteroids before finally returning to Earth. The performance index to be maximized here is a function of the final mass of and the time of stay on the asteroids. However, in this work, the total flight time and the time of stay were not considered; instead, only the propellant quasi-optimal visit sequence for the given asteroids and visit requirements was studied. Thus, a trajectory-generation example for 3 out of 140 asteroids was used for the calculation. The departure Modified Julian Date (MJD) is 58,091, the propulsive acceleration limit is set to $a_{\max} = 7.5 \times 10^{-5}$ m/s^2, the initial mass of the spacecraft is 2,000 kg, and the specific impulse is 3,000 s [29]. The Fourier terms are $n_r = 6$, $n_\theta = 6$, $n_z = 8$, and m and n_{App} are 120 and 600, respectively [18]. All tests were conducted on a Core i7 3.40 GHz processor with Windows 7 and run on MATLAB R2015b.

Zhu et al. [24] reduced 140 asteroids to 16 by excluding the asteroids with inclinations greater than 0.06 rad and eccentricities

Table 1
Feasible visiting sequences for 16 asteroids.

Number	Asteroid 1	Asteroid 2	Asteroid 3	ΔV/(km/s)
1	61	96	49	16.0484
2	61	96	88	18.3284
3	88	49	76	13.7513
4	88	49	96	13.8286
5	88	76	49	11.8380
6	88	76	96	12.3532
7	88	96	49	9.9415
8	88	96	76	13.9597
9	96	49	76	15.9669
10	96	49	88	12.7163
11	96	76	49	12.6610
12	96	76	88	12.6053
13	96	88	49	9.0787
14	96	88	76	12.1710

greater than 0.17, and selected 3 of the 16 asteroids for exploration. Hence, this study first calculates the quasi-optimal solution from these 16 asteroids. Data on these 16 asteroids can be found in Ref. [24].

5.1. Traversal algorithm

The goal of the traversal algorithm is to attempt all the visiting sequences once, discard those that do not satisfy the constraints, and record those that do satisfy the constraints. It can find all the feasible visiting sequences; however, the computational time taken to find the sequences is too long. This work uses the traversal algorithm for comparison, and Table 1 is the result of its operation. As can be seen from the table, the optimal visiting sequence is Earth → 96 → 88 → 49 → Earth, and the smallest ΔV is 9.0787 km/s. The overall computation time is 1,515.6001 s. The flight time from Earth to the asteroid 96 is 723.6 days, from asteroid 96 to asteroid 88 is 820.4 days, from asteroid 88 to asteroid 49 is 621.6 days, and from asteroid 49 to Earth is 902.6 days, and the total flight time is 3,068.2 days.

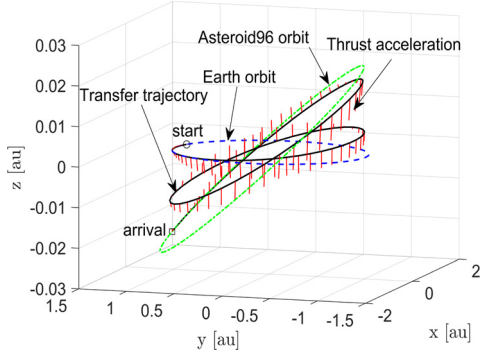

Fig. 6. Trajectory from Earth to Asteroid 96.

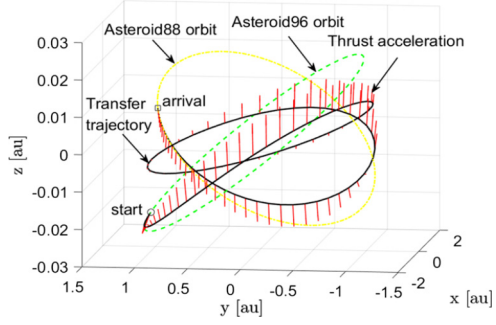

Fig. 7. Trajectory from Asteroid 96 to Asteroid 88.

The quasi-optimal visiting sequence has been determined, and the transfer trajectories determined by FFS are illustrated in Figs. 6–9. Figs. 10–13 show the propulsive acceleration vector and its three components.

5.2. Greedy algorithm

A greedy algorithm is a fast method to determine a local optimal solution; however, it has a small probability of finding the quasi-optimal solution. In this work, a greedy algorithm was used to quickly determine a local optimal visiting sequence.

First, the first asteroids in the visiting sequence are chosen. The trajectories from Earth to 16 asteroids are generated, and then the asteroid with the lowest performance index is chosen as the asteroid visited first. The trajectories from that asteroid to the remaining asteroids are then generated separately, and the asteroid with the best performance index is chosen as the asteroid visited second. The process is repeated in iterations until the spacecraft visits three asteroids and returns to Earth. In this process, if it is impossible to generate a trajectory from an asteroid to any other asteroid under the current conditions, it is abandoned, and the previous step is revisited. The asteroid with the next smallest performance index is thus chosen for the following orbit transfer. Finally, the calculation results of the greedy algorithm are Earth → 88 → 96 → 49 → Earth, and the smallest ΔV is 9.9415 km/s. The overall computation time is 327.0277 s.

5.3. Tree search algorithm with trimming strategy

The tree search algorithm with trimming strategy is necessary to avoid the unnecessary traversal process due to some judgment conditions, which is to cut off a few “branches” of the search tree. The core problem of applying a trimming strategy is to design an appropriate trimming judgment condition.

When designing judgment methods, certain principles must be followed. These principles for trimming are correctness, accuracy,

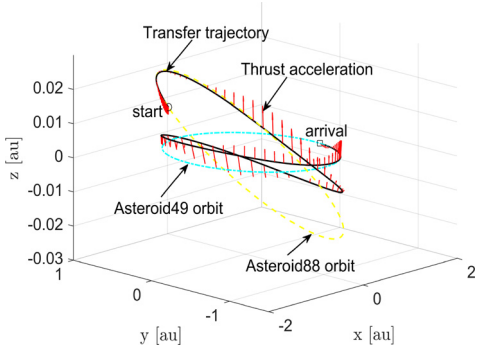

Fig. 8. Trajectory from Asteroid 88 to Asteroid 49.

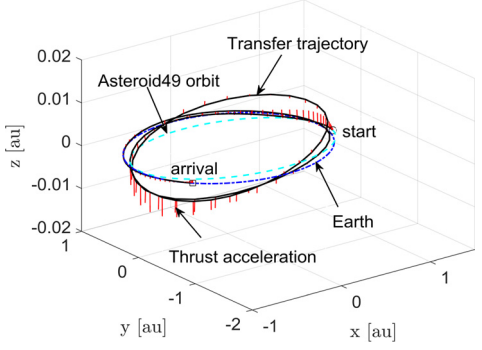

Fig. 9. Trajectory from Asteroid 49 to Earth.

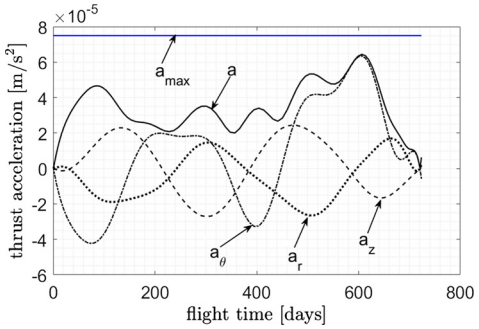

Fig. 10. Propulsive acceleration of the trajectory from Earth to Asteroid 96.

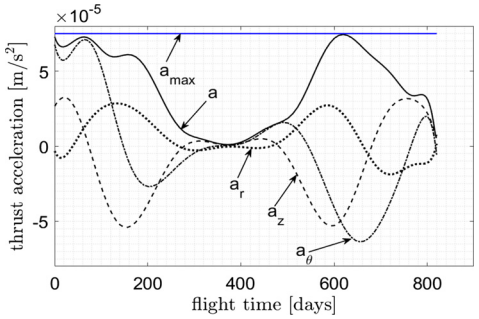

Fig. 11. Propulsive acceleration of the trajectory from Asteroid 96 to Asteroid 88.

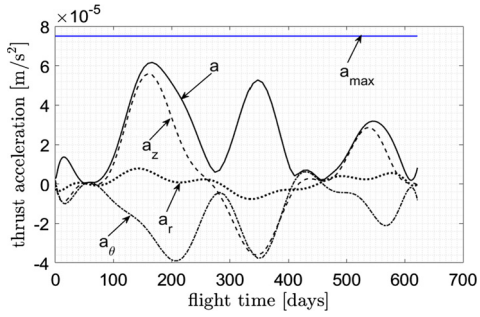

Fig. 12. Propulsive acceleration of the trajectory from Asteroid 88 to Asteroid 49.

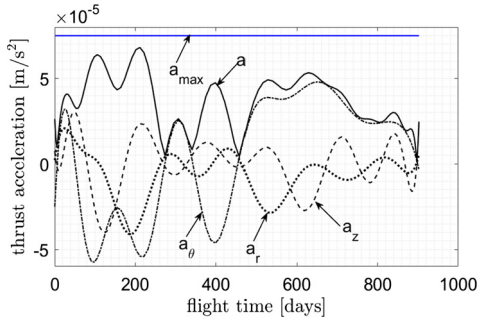

Fig. 13. Propulsive acceleration of the trajectory from Asteroid 49 to Earth.

and high efficiency. It is very important to find a balance between optimization and efficiency, so as to reduce the time complexity of the program as much as possible.

In order to ensure a reasonable trimming condition, the solution of the greedy algorithm was chosen as the trimming condition in this study. When the total performance of any one or several times was larger than that of the greedy algorithm, the branch is cut off and the downward search was discontinued. The calculation results of the tree search algorithm with trimming strategy are Earth $\rightarrow$ 96 $\rightarrow$ 88 $\rightarrow$ 49 $\rightarrow$ Earth, and the smallest ΔV is 9.0787 km/s. The overall computation time is 1,377.7018 s.

5.4. Monte-Carlo tree search algorithm

In Eq. (17), the formula of UCB consists of two parts: the first part is $-\overline{V_i}$, which can be seen as an estimate of the average revenue per time of the subnode. The second item involves the calculation of the number of visits to a node. The less visited the node, the larger the value of this part. The parameter c in Eq. (17) is the compromise coefficient between the average revenue and the number of explorations. Therefore, the parameter c should have a suitable value to make the algorithm more effective and accurate.

In the MCTS algorithm, because of the random search process, when the number of iterations is small, the results of the algorithm may be different each time for the same number of iterations. When the number of iterations reaches a certain value, the optimal solution can be obtained with a greater probability. Therefore, in this study, for each specific c value and iteration step, 100 repeated calculations were carried out under the same conditions, and the probability that the algorithm can attain the optimal solution under different conditions was obtained. Corresponding to the different values of c, Fig. 14 presents the percentage of finding the optimal visiting sequence with an increase in the number of iterations. The curve of change of the computational time with the increase in the number of iterations is shown in Fig. 15.

In Fig. 14, as the value of c gradually increases from 0.4 to 1.2, the MCTS algorithm can no longer obtain the optimal solution with sufficiently high probability even with the increase in the number of iterations. However, with the increase of c value and the number of iterations, the probability of obtaining the optimal solution increases. When the value of c increases from 1.3 to 2, the MCTS algorithm can obtain the optimal solution with a large enough probability by increasing the number of iterations. However, when the value of c increases gradually from 1.3, and the number of iteration steps is small, the calculation results of the MCTS algorithm become worse with the increase in the c value, and the probability of finding the optimal solution decreases. This result can be explained in Eq. (17). The smaller the c value is, the smaller the impact of the number of searches on the UCB value is, which will lead to the difficulty of "jumping out" once the search enters the local optimal solution. The larger the c value is, the greater the impact of the search time on the UCB value is, the larger the search dimension is, and the more obvious the "jump out" is. Thus, when the c value increases, the probability of obtaining the optimal solution in the early stage will decrease.

In Fig. 15, for the same c value, the average computational time increases with the number of iterations; for the same iteration step, the average computational time also increases with the increase in the c value because the increasing c value increases the search dimension. In order to obtain the optimal solution with higher probability and at a shorter time, we can choose a c value of 1.3 and an iteration number of 15. Based on the results of 100 calculations under this condition, the probability of obtaining the quasi-optimal solution is 100%. The computation time is 1,197.8493 s, which is relatively short.

5.5. Results comparison

The results of the four algorithms are shown in Table 2. It is seen that despite being the fastest method, the greedy algorithm cannot obtain the quasi-optimal solution, whereas the other three algorithms can. In addition, if the computation time of the trimming condition (i.e. the result of the greedy algorithm) is considered, then the computation time of the trimming algorithm should be the longest. Comparing the computational time of the traversal algorithm with that of MCTS, it is seen that the shortest computa-

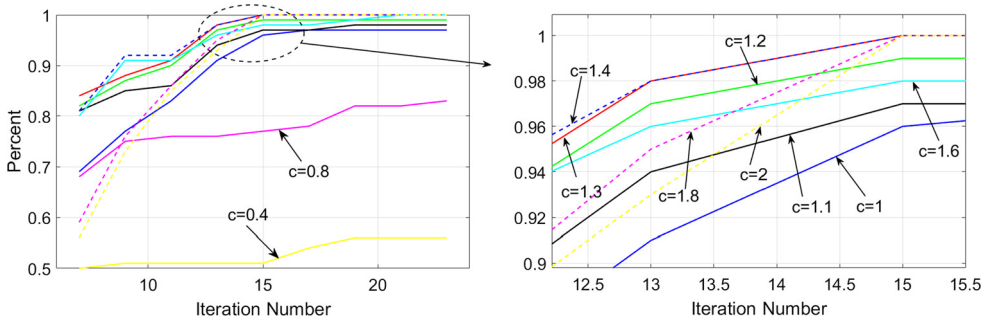

Fig. 14. Percentage chance of finding the optimal visiting sequence corresponding to different c with an increase in the number of iterations.

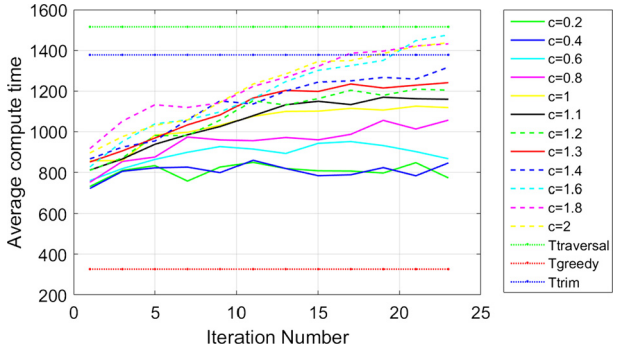

Fig. 15. Curve of change in computational time corresponding to different c with an increase in the number of iterations.

Table 2
Results comparison.

Algorithms	Sequences	ΔV/(km/s)	Computation time/(s)
Traversal	96→88→49	9.0787	1,515.6001
Greedy	88→96→49	9.9415	327.0277
Trimming	96→88→49	9.0787	1,377.7018
MCTS	96→88→49	9.0787	1,197.8493

tional time of MCTS is approximately 79% of that of the traversal algorithm when the quasi-optimal solution of MCTS is to be obtained with 100% probability.

Decreasing the number of iterations can reduce the time needed to obtain the results. For example, when the number of iterations is reduced to 9, the probability of obtaining the optimal solution is 88%, and the probability of obtaining the suboptimal solution is 12%. The difference between the value of the suboptimal solution and the optimal solution is 9.5%, and the computation time is 71% of that of the traversal algorithm.

6. Conclusion

This paper presents a method for the rapid preliminary design of visiting sequences and trajectory optimization in multi-asteroid exploration missions for low-thrust trajectories, based on MCTS and FFS. The FFS method is used to design reasonable initial three-dimensional trajectories to enable a more accurate optimization solver in a short time, and all the constraints associated with low thrust-propelled trajectories are satisfied. The MCTS algorithm is used to select the quasi-optimal visiting sequences for multi-asteroid exploration missions. In order to verify the effectiveness of the proposed method for these missions, the asteroids in GTOC-3 were selected for calculations. Compared to the traversal algorithm, the greedy algorithm, and the trimming strategy, MCTS obtained the quasi-optimal visiting sequence with the least computational time. Based on the results of 100 calculations under this condition, the probability of obtaining a quasi-optimal visiting sequence is 100%. The computation time is 1,197.8493 s, which is 79% of the computational time required by the traversal algorithm. When the number of iterations is reduced to 9, MCTS takes 71% of the computation time of the traversal algorithm to obtain the quasi-optimal solution with a probability of 88% and the suboptimal solution with a probability of 12%, and the difference between the two solutions is 9.5%. Therefore, the MCTS algorithm can obtain the quasi-optimal visiting sequence with least computation time among the four algorithms. This is of great significance for the rapid feasibility assessment of hundreds of flight scenarios at the preliminary mission design stage.

Declaration of Competing Interest

There is no competing interest.

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China under Grant Nos. 11702072 and 11672093, the Innovation Fund of the Shanghai Academy of Spaceflight Technology (SAST) under Grant No. SAST2016039, the China Postdoctoral Science Foundation under Grant No. 2017M611372, the Heilongjiang Postdoctoral Fund under Grant No. LBH-Z16082, and the innovation fund of Harbin Institute of Technology under Grant No. 30620170018.

References

[1] K. Wang, B. Zhang, Multiobjective trajectory optimization for a suborbital spaceplane using directed search domain approach, Aerosp. Sci. Technol. 77 (2018) 713–724, https://doi.org/10.1016/j.ast.2018.04.018.
[2] D.M. Novak, M. Vasile, Improved shaping approach to the preliminary design of low-thrust trajectories, J. Guid. Control Dyn. 34 (1) (2011) 128–147, https://doi.org/10.2514/1.50434.
[3] J.T. Betts, Survey of numerical methods for trajectory optimization, J. Guid. Control Dyn. 21 (2) (1998) 193–207, https://doi.org/10.2514/2.4231.
[4] H. Shang, X. Wu, D. Qiao, X. Huang, Parameter estimation for optimal asteroid transfer trajectories using supervised machine learning, Aerosp. Sci. Technol. 79 (2018) 570–579, https://doi.org/10.1016/j.ast.2018.06.002.
[5] L. Niccolai, A.A. Quarta, G. Mengali, Analytical solution of the optimal steering law for non-ideal solar sail, Aerosp. Sci. Technol. 62 (2017) 11–18, https://doi.org/10.1016/j.ast.2016.11.031.
[6] Q. Hu, J. Xie, X. Liu, Trajectory optimization for accompanying satellite obstacle avoidance, Aerosp. Sci. Technol. 82–83 (2018) 220–233, https://doi.org/10.1016/j.ast.2018.08.033.
[7] A.E. Petropoulos, J.M. Longuski, Shape-based algorithm for the automated design of low-thrust, gravity-assist trajectories, J. Spacecr. Rockets 41 (5) (2004) 787–796, https://doi.org/10.2514/1.13095.
[8] P. De Pascale, M. Vasile, Preliminary design of low-thrust multiple gravity-assist trajectories, J. Spacecr. Rockets 43 (5) (2006) 1069–1076, https://doi.org/10.2514/1.19646.
[9] B.J. Wall, B.A. Conway, Shape-based approach to low-thrust rendezvous trajectory design, J. Guid. Control Dyn. 32 (1) (2009) 95–101, https://doi.org/10.2514/1.36848.
[10] B. Wall, Shape-based approximation method for low-thrust trajectory optimization, AIAA/AAS Astrodynam. Spec. Conf. Exhibit (2008) 2008-6616, https://doi.org/10.2514/6.2008-6616.

[11] C. Xie, G. Zhang, Y. Zhang, Simple shaping approximation for low-thrust trajectories between coplanar elliptical orbits, J. Guid. Control Dyn. 38 (12) (2015) 2448–2455, https://doi.org/10.2514/1.G001209.
[12] D.J. Gondelach, R. Noomen, Hodographic-shaping method for low-thrust interplanetary trajectory design, J. Spacecr. Rockets 52 (3) (2015) 728–738, https://doi.org/10.2514/1.A32991.
[13] C. Xie, G. Zhang, Y. Zhang, Shaping approximation for low-thrust trajectories with large out-of-plane motion, J. Guid. Control Dyn. 39 (12) (2016), https://doi.org/10.2514/1.G001795.
[14] K. Zeng, Y. Geng, B. Wu, Shape-based analytic safe trajectory design for spacecraft equipped with low-thrust engines, Aerosp. Sci. Technol. 62 (2017) 87–97, https://doi.org/10.1016/j.ast.2016.12.006.
[15] A. Peloni, B. Dachwald, M. Ceriotti, Multiple near-Earth asteroid rendezvous mission: solar-sailing options, Adv. Space Res. 62 (8) (2018) 2084–2098, https://doi.org/10.1016/j.asr.2017.10.017.
[16] O. Abdelkhalik, E. Taheri, Approximate on-off low-thrust space trajectories using Fourier series, J. Spacecr. Rockets 49 (5) (2012) 962–965, https://doi.org/10.2514/1.60045.
[17] E. Taheri, O. Abdelkhalik, Fast initial trajectory design for low-thrust restricted-three-body problems, J. Guid. Control Dyn. 38 (11) (2015) 1–15, https://doi.org/10.2514/1.G000878.
[18] E. Taheri, O. Abdelkhalik, Initial three-dimensional low-thrust trajectory design, Adv. Space Res. 57 (3) (2016) 889–903, https://doi.org/10.1016/j.asr.2015.11.034.
[19] E. Taheri, I. Kolmanovsky, E. Atkins, Shaping low-thrust trajectories with thrust-handling feature, Adv. Space Res. 61 (3) (2018) 879–890, https://doi.org/10.1016/j.asr.2017.11.006.
[20] Y. Ulybyshev, Continuous thrust orbit transfer optimization using large-scale linear programming, J. Guid. Control Dyn. 30 (2) (2007) 427–436, https://doi.org/10.2514/1.22642.
[21] J.T. Betts, Very low-thrust trajectory optimization using a direct SQP method, J. Comput. Appl. Math. 120 (1) (2000) 27–40, https://doi.org/10.1016/S0377-0427(00)00301-0.
[22] A.D. Olds, C.A. Kluever, M.L. Cupples, Interplanetary mission design using differential evolution, J. Spacecr. Rockets 44 (5) (2007) 1060–1070, https://doi.org/10.2514/1.27242.
[23] D.G. Hull, Conversion of optimal control problems into parameter optimization problems, J. Guid. Control Dyn. 20 (1) (1997) 57–60, https://doi.org/10.2514/2.4033.
[24] K. Zhu, F. Jiang, J. Li, H. Baoyin, Trajectory optimization of multi-asteroids exploration with low thrust, Trans. Jpn. Soc. Aeronaut. Space Sci. 52 (175) (2009) 47–54, https://doi.org/10.2322/tjsass.52.47.
[25] M. Yu, C. Li, Robust adaptive iterative learning control for discrete-time nonlinear systems with time-iteration-varying parameters, IEEE Trans. Syst. Man Cybern. Syst. 47 (7) (2017) 1737–1745, https://doi.org/10.1109/TSMC.2017.2677959.
[26] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G.V.D. Driessche, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489, https://doi.org/10.1038/nature16961.
[27] D. Silver, T. Hubert, S. Schrittwieser, et al., A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play, Science 362 (6419) (2018) 1140–1144, https://doi.org/10.1126/science.aar6404.
[28] X. Hu, S. Gong, Flexibility influence on passive stability of a spinning solar sail, Aerosol Sci. Technol. 58 (2016) 60–70, https://doi.org/10.1016/j.ast.2016.12.006.
[29] https://sophia.estec.esa.int/gtoc_portal/?page_id=17.