Robotica: (2020) volume 38, pp. 957–982. © Cambridge University Press 2019
doi:10.1017/S0263574719001176

Point-to-Point Motion Planning of a Free-Floating Space Manipulator Using the Rapidly-Exploring Random Trees (RRT) Method

Tomasz Rybus*

Electronics Faculty, Wrocław University of Science and Technology, Janiszewskiego 11/17, 50-372 Wrocław, Poland and Space Mechatronics and Robotics Laboratory, Space Research Centre (CBK PAN), Bartycka 18a, 00-716 Warsaw, Poland

(Accepted June 24, 2019. First published online: July 24, 2019)

SUMMARY

It is usually proposed to use a robotic manipulator for performing on-orbit capture of a target satellite in the planned active debris removal and on-orbit servicing missions. Control of the satellite-manipulator system is challenging because motion of the manipulator influences position and orientation of the chaser satellite. Moreover, the trajectory selected for the capture manoeuvre must be collision-free. In this article, we consider the case of a nonredundant manipulator mounted on a free-floating satellite. We propose to use the bi-directional rapidly-exploring random trees (RRT) algorithm to achieve two purposes: to plan a collision-free manipulator trajectory that, at the same time, will result in a desired change of the chaser satellite orientation. Several improvements are introduced in comparison to the previous applications of the RRT method for manipulator mounted on a free-floating satellite. Feasibility of the proposed approach is demonstrated in numerical simulations performed for the planar case in which the chaser satellite is equipped with a 2-DoF (Degree of Freedom) manipulator. The obtained results are analysed and compared with the results obtained from collision-free trajectory planning methods that do not allow to set the desired final orientation of the chaser satellite.

KEYWORDS: Space robotics; RRT algorithm; Trajectory planning.

1. Introduction

New, ambitious unmanned space missions that are currently planned include active debris removal (ADR) and on-orbit servicing (OOS). The purpose of OOS missions is to prolong operational lifetime of satellites by performing on-orbit refuelling, component replacement and repairs.[1] Studies show that such missions could be economically feasible.[2] The goal of the ADR missions is to remove from orbit large space debris (e.g. spent rocket stages, defunct satellites).[3] Studies show that ADR missions will be necessary to stop the growth of the space debris population in low Earth orbit.[4] The European Space Agency is currently preparing the e.Deorbit mission to demonstrate technologies required for ADR.[5] ADR missions are also considered for the geostationary orbit.[6]

It is usually proposed to use a robotic manipulator for performing on-orbit capture of a target satellite.[7–9] High level of on-board autonomy is required for the manipulator-equipped chaser satellite.[10] The capture manoeuvre cannot be performed entirely in the teleoperation mode (by ground controllers) because of the communication link latency and possible communication dropouts.[11] Thus, in the proposed ADR and OOS missions, efficient methods for the manipulator trajectory planning and control must be implemented on the chaser satellite on-board computer.

* Corresponding author. E-mail: trybus@cbk.waw.pl

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

Control of the manipulator mounted on a relatively small satellite is a challenging task, because motion of the manipulator during the orbital capture manoeuvre influences the position and orientation of the chaser satellite. One possible approach is to use thrusters and reaction wheels to compensate reaction torques and reaction forces induced by the motion of the manipulator.[12] There are several studies devoted to the problem of coordinated control of the satellite-manipulator system. In the approach presented in ref. [13], the robot control system estimates the angular momentum which the manipulator will produce. The satellite attitude control system compensates the manipulator's reaction by the feed forward momentum control. A control scheme presented by Egeland and Sagli allows precise control of the manipulator end-effector and gross positioning of the satellite (using reaction wheels and thrusters).[14] The coordination control based on the transposed Jacobian-type controller with inertial feedback is also proposed.[15] Full control of the manipulator-equipped chaser satellite position and orientation in the capture phase is considered in the e.Deorbit mission.[16] However, using the control system of the satellite to compensate the influence of the manipulator motion may be difficult, because reaction torques and reaction forces induced by the manipulator may have high values and may change rapidly. Thus, designing suitable control system is a challenging task.[17] Moreover, undesirable and dangerous interactions between the control system of the manipulator and control system of the satellite may occur. Therefore, in some studies, it is proposed to switch off the control system of the satellite in the capture phase.[18,19] In such case, the satellite-manipulator system is in a free-floating state.[20]

Trajectory, selected for the capture manoeuvre, must be collision-free. Taking into account obstacles in the manipulator workspace is a challenging task due to the possible relative motion between the two involved satellites and due to the complicated shape of the satellites.[21] Methods for planning collision-free trajectories for fixed-base manipulators are in development since 1970's,[22] but in the field of space robotics, this is a relatively new area of studies. Review of various approaches for collision-free trajectory planning of space manipulators is presented in ref. [23]. These approaches include methods based on non-linear optimisation,[24,25] methods based on the artificial potential field (APF),[26,27] methods based on the A* algorithm[28,29] and methods based on the rapidly-exploring random trees (RRT) algorithm.[30,31] Combinations of different methods are also proposed. The APF method can be combined with a non-linear optimisation[32] or with a graph-search algorithm.[33]

In this article, we propose to use a method based on the bi-directional RRT algorithm to achieve two purposes: to plan a collision-free manipulator trajectory that, at the same time, will result in a desired change of the chaser satellite orientation. To the best of our knowledge, this is the first approach that not only allows to solve the obstacle avoidance problem, but also allows to control the chaser satellite orientation for the case of a nonredundant manipulator mounted on a free-floating satellite. Although the RRT method is a well-known approach, in this article, we present specific application of this method for a new and challenging task that was not previously solved. Moreover, several improvements are introduced in comparison to the previous applications of the RRT method for manipulator mounted on a free-floating satellite (i.a. reduction of the dimension of the state vector used in the tree construction). Presented results of numerical simulations prove the feasibility of the proposed method. The approach presented in this article has a practical significance in the context of the planned OOS and ADR missions.

The article is organised as follows. The manipulator trajectory planning methods that allow to obtain the desired changes of the chaser satellite orientation are reviewed in Section 2. The equations of the satellite-manipulator dynamics are presented in Section 3. The proposed trajectory planning method is described in Section 4. Results of numerical simulations performed for verification of the proposed approach are shown in Section 5, where the trajectory planning method based on the bi-directional RRT algorithm is compared with two other methods: the method based on the basic RRT algorithm (one-directional) and the APF method. Section 6 contains the discussion of the obtained results. Detailed analysis of the proposed approach is shown in Section 7. The article concludes with a short summary given in Section 8.

2. Review of Trajectory Planning Methods for a Manipulator Mounted on a Free-Floating Satellite

The problem of simultaneous control of a manipulator and chaser satellite orientation by the use of manipulator joints only has been extensively studied in the past. In this section, we review various methods that are proposed for this purpose.

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

To solve the problem of the influence of manipulator motion on the state of the free-floating chaser satellite, it was proposed to use a second manipulator (balance arm) to compensate the reaction torques and reaction forces induced by the motion of the manipulator.[34] In practice, such solution may not be feasible due to increased cost and complexity of the satellite-manipulator system. However, even if a single manipulator is used, the influence of its motion on the state of the chaser satellite can be minimised or even completely eliminated by selecting specific trajectory of the manipulator joints. As pointed out by Oda,[35] in practical applications, it may be very important to select such trajectory, which will guarantee no changes of the satellite attitude, because changes of the attitude could disturb communication link or generation of power from the solar panels (changes of the satellite position are far less problematic). Several practical tasks can be achieved with reactionless manipulation.[36]

If the manipulator is redundant, then the additional DoF of the manipulator can be exploited to achieve desired motion of the chaser satellite, for example, to keep the satellite orientation fixed or to change it to a desired value. An algorithm (based on the Virtual Manipulator concept[37]) that allows such simultaneous control of the manipulator end-effector and orientation of the satellite is presented by Vafa.[38] One of the manipulators considered in his analysis has 9 DoFs: 6 DoFs are used to move the end-effector through a desired trajectory with a desired orientation, while the additional 3 DoFs are used to keep constant orientation of the satellite. A similar approach was demonstrated in experiments performed on a planar air-bearing microgravity simulator.[39] In such test-bed, the motion is limited to one plane. In these experiments, a satellite mock-up with a 3-DoF manipulator was used. The end-effector was following a desired trajectory but without any constraints on its orientation. Thus, the 3-DoF manipulator was redundant and one additional DoF was used to control the orientation of the satellite mock-up. Another approach for planning the trajectory that does not cause changes in satellite attitude is presented by Zhang et al.[40]

Reactionless trajectory planning is much more difficult for nonredundant manipulators. An approach, based on the reaction null space (RNS) method, is proposed by Piersigilli et al. for generating trajectories that allow grasping of the target satellite which is rotating with a constant angular velocity.[41] The orientation of the chaser satellite, equipped with a nonredundant manipulator, is not changed during realisation of these trajectories. The proposed method was tested in numerical simulations performed for a planar case, in which the satellite is equipped with a 2-DoF manipulator. The RNS-based reactionless manipulation was successfully tested in experiments performed during the ETS-VII orbital technology demonstration mission.[42] In this mission, the chaser satellite was equipped with a nonredundant 6-DoF manipulator and a set of specific manipulator trajectories that do not influence the orientation of the satellite was realised.

A method for trajectory planning using the Lyapunov function is proposed by Nakamura and Mukherjee.[43] This method is based on the bi-directional approach and allows to achieve desired positions of manipulator joints and desired attitude of the chaser satellite. Results of numerical simulations are presented for a satellite equipped with a 6-DoF nonredundant manipulator. Another approach is based on the basis algorithm, which is used to obtain near-optimal solutions of the trajectory planning problem.[44] In the numerical simulation, 3-DoF spatial manipulator is considered and it is shown that the proposed method can be used to achieve desired final position of the end-effector and desired final attitude of the chaser satellite. An approach based on the particle swarm optimisation is proposed by Xu et al.[45] Another trajectory planning method proposed for satellite-mounted manipulators is based on mapping the nonholonomic constraint to a space where it can be satisfied trivially.[46] This method uses polynomials to drive the system to a desired final configuration in a given time. Simultaneous manipulator end-effector and chaser attitude control is possible using manipulator actuators only. Additional requirements such as obstacle avoidance can be fulfilled using higher order polynomials.[47,48]

The RRT algorithm[49] is another approach that can be used for planning manipulator trajectory and obtaining the desired change of the chaser satellite attitude. Use of the RRT algorithm is an interesting alternative to the methods reviewed above, as this approach also allows to plan a collision-free trajectory that takes into account obstacles in the manipulator workspace.

3. Dynamics of the Free-Floating Satellite-Manipulator System

In our study, we are considering a manipulator with n rotational joints. This manipulator is mounted on a free-floating satellite. We assume that there are no external torques and external forces acting on the system (in the short time scale of the orbital capture manoeuvre disturbances, such as the

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

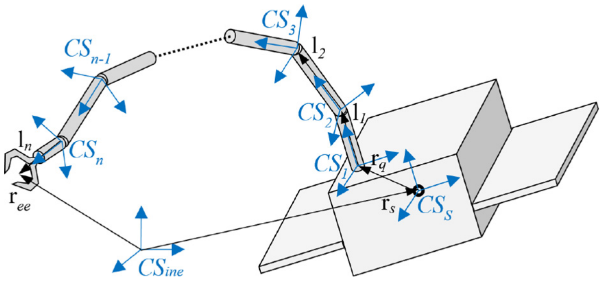

Fig. 1. A schematic view of the chaser satellite equipped with the n-DoF manipulator.

gravity gradient, can be neglected). Control system of the chaser satellite is switched off during the motion of the manipulator. We use equations presented by Seweryn and Banaszkiewicz[50] and Rybus et al.[51] These equations are based on the generalised Jacobian matrix introduced by Umetani and Yoshida.[52] Schematic view of the chaser satellite equipped with the n-DoF manipulator is shown in Fig. 1. Equations presented in this section are given in the inertial reference frame denoted by CS_{ine}. Coordinate frame denoted by CS_S is attached to the chaser satellite and located at its centre of mass. Coordinate frame attached to the i-th link of the manipulator and located at the i-th joint is denoted by CS_i. The Z-axis of CS_i coincides with the i-th link of the manipulator.

In our approach, we use the generalised coordinates defined as[53]

$$\mathbf{q}_p = \begin{bmatrix} \mathbf{r}_s \\ \boldsymbol{\Theta}_s \\ \boldsymbol{\theta} \end{bmatrix} \tag{1}$$

where $\mathbf{r}_s$ denotes the position of the chaser satellite centre of mass, $\boldsymbol{\Theta}_s$ denotes the orientation of the satellite (described by Euler angles in ZYX convention), while $\boldsymbol{\theta}$ is an n-dimensional vector that contains positions of manipulator joints. The state of the satellite-manipulator system is fully described by the following state vector:

$$\mathbf{x} = \begin{bmatrix} \mathbf{q}_v \\ \mathbf{q}_p \end{bmatrix} \tag{2}$$

where $\mathbf{q}_v$ is defined as

$$\mathbf{q}_v = \begin{bmatrix} \mathbf{v}_s \\ \boldsymbol{\omega}_s \\ \dot{\boldsymbol{\theta}} \end{bmatrix} = \begin{bmatrix} \frac{d\mathbf{r}_s}{dt} \\ \mathbf{T}_{\boldsymbol{\Theta}}^{-1} \frac{\boldsymbol{\Theta}_s}{dt} \\ \frac{d\boldsymbol{\theta}}{dt} \end{bmatrix} \tag{3}$$

The matrix $\mathbf{T}_{\boldsymbol{\Theta}}$ transforms the angular velocity into time derivatives of Euler angles. The kinetic energy of the satellite-manipulator system is expressed as

$$T = \frac{1}{2} \begin{bmatrix} \mathbf{v}_s \\ \boldsymbol{\omega}_s \\ \dot{\boldsymbol{\theta}} \end{bmatrix}^T \begin{bmatrix} \mathbf{A} & \mathbf{B} & \mathbf{D} \\ \mathbf{B}^T & \mathbf{E} & \mathbf{F} \\ \mathbf{D}^T & \mathbf{F}^T & \mathbf{N} \end{bmatrix} \begin{bmatrix} \mathbf{v}_s \\ \boldsymbol{\omega}_s \\ \dot{\boldsymbol{\theta}} \end{bmatrix} \tag{4}$$

The submatrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$ and $\mathbf{N}$ are defined as

$$\mathbf{A} = \left(m_s + \sum_{i=1}^{n} m_i \right) \mathbf{I} \tag{5}$$

$$\mathbf{B} = \left(m_s + \sum_{i=1}^{n} m_i \right) \tilde{\mathbf{r}}_{s\_g} = \sum_{i=1}^{n} m_i \tilde{\mathbf{r}}_{i\_s} \tag{6}$$

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

$$\mathbf{D} = \sum_{i=1}^{n} m_i \mathbf{J}_{Ti} \tag{7}$$

$$\mathbf{E} = \mathbf{I}_s + \sum_{i=1}^{n} \left(\mathbf{I}_i + m_i \tilde{\mathbf{r}}_{i\_s}^T \tilde{\mathbf{r}}_{i\_s} \right) \tag{8}$$

$$\mathbf{F} = \sum_{i=1}^{n} \left(\mathbf{I}_i \mathbf{J}_{Ri} + m_i \tilde{\mathbf{r}}_{i\_s} \mathbf{J}_{Ti} \right) \tag{9}$$

$$\mathbf{N} = \sum_{i=1}^{n} \left(\mathbf{J}_{Ri}^T \mathbf{I}_i \mathbf{J}_{Ri} + m_i \mathbf{J}_{Ti}^T \mathbf{J}_{Ti} \right) \tag{10}$$

where

$$\mathbf{r}_{s\_g} = \mathbf{r}_{CM} - \mathbf{r}_s \tag{11}$$

$$\mathbf{r}_{i\_s} = \mathbf{r}_i - \mathbf{r}_s \tag{12}$$

In the above equations, m_s and $\mathbf{I}_s$ are the mass and inertia matrix of the satellite, respectively, m_i and $\mathbf{I}_i$ are the mass and inertia matrix of the i-th link of the manipulator, vector $\mathbf{r}_{CM}$ denotes the mass centre of the satellite-manipulator system, while vector $\mathbf{r}_i$ denotes the mass centre of the i-th link of the manipulator, $\mathbf{J}_{Ti}$ is the translational component of the manipulator Jacobian matrix, while $\mathbf{J}_{Ri}$ is the rotational component of this matrix, $\mathbf{I}$ denotes the identity matrix, the $\sim$ symbol denotes a matrix which is the equivalent of a vector cross product.

A potential energy of the system is neglected, because the satellite is in the state of free fall. Substituting (4) into the Lagrange equation, we obtain the generalised equations of motion

$$\mathbf{Q} = \mathbf{M}\left(\mathbf{q}_p\right) \dot{\mathbf{q}}_v + \mathbf{C}\left(\mathbf{q}_v, \mathbf{q}_p\right) \mathbf{q}_v \tag{13}$$

where vector $\dot{\mathbf{q}}_v$ denotes the first derivative of $\mathbf{q}_v$ with respect to time, while $\mathbf{Q}$ denotes the vector of generalised forces

$$\mathbf{Q} = \begin{bmatrix} \mathbf{F}_s \\ \mathbf{H}_s \\ \mathbf{u} \end{bmatrix} \tag{14}$$

The external forces and the external torques acting on the satellite are denoted by $\mathbf{F}_s$ and $\mathbf{H}_s$, respectively, while $\mathbf{u}$ is an n-dimensional vector of control torques applied at manipulator joints. As there are no external disturbances and the control system of the satellite is switched off , we assume $\mathbf{F}_s = 0$ and $\mathbf{H}_s = 0$. We consider a case in which the linear and angular momentum of the satellite-manipulator system is zero. However, as explained in ref. [51], the general approach presented in this section can be used for systems with a constant non-zero linear and angular momentum, as well as for systems subject to external torques and forces (in such case, the momentum and angular momentum change, e.g. due to actions of cold-gas thrusters mounted on the chaser satellite[54]).

The mass matrix, $\mathbf{M}$, is defined as

$$\mathbf{M}\left(\mathbf{q}_p\right) = \begin{bmatrix} \mathbf{A} & \mathbf{B} & \mathbf{D} \\ \mathbf{B}^T & \mathbf{E} & \mathbf{F} \\ \mathbf{D}^T & \mathbf{F}^T & \mathbf{N} \end{bmatrix} \tag{15}$$

while the components of the Coriolis matrix, $\mathbf{C}$, can be computed from

$$C_{ij} = \sum_{k=1}^{n} \left(\frac{d}{dq_k} m_{ij} - \frac{1}{2} \frac{d}{dq_i} m_{jk} \right) \tag{16}$$

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

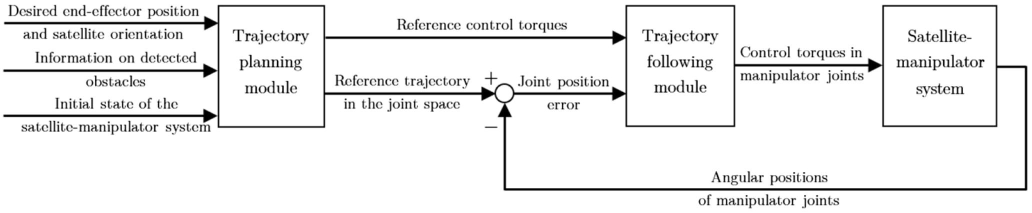
Desired end-effector position and satellite orientation
Information on detected obstacles
Initial state of the satellite-manipulator system
Trajectory planning module
Reference control torques
Reference trajectory in the joint space
+
−
Joint position error
Trajectory following module
Control torques in manipulator joints
Satellite-manipulator system
Angular positions of manipulator joints

Fig. 2. Structure of the control system for manipulator mounted on the chaser satellite.

where m_{ij} denotes the element lying in the i-th row and j-th column of the mass matrix $\mathbf{M}$. Indices i, j and k in (16) take values from 1 to n. Finally, we obtain the following set of equations that describe dynamics of the satellite-manipulator system in the free-floating state:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{M}^{-1}\left(\mathbf{Q} - \mathbf{C}\mathbf{q}_v\right) \\ \mathbf{q}_v \end{bmatrix} \tag{17}$$

4. Trajectory Planning with the RRT Algorithm

As explained in ref. [51], the control system of the manipulator mounted on the chaser satellite can be divided into two separate modules: (i) the trajectory planning module and (ii) the trajectory following module. The first module is responsible for finding a suitable trajectory of the manipulator, while the second module is a closed-loop controller responsible for ensuring realisation of this trajectory. The general structure of such control system is presented in Fig. 2. The trajectory planning is performed just before the execution of the capture manoeuvre. The goal of the trajectory planning algorithm presented in this section is to find a collision-free trajectory of the manipulator from a given initial state of the satellite-manipulator system to a final state in which the end-effector is in a desired position (e.g. in a position that allows grasping of the target satellite). This trajectory must also provide a desired change of the chaser satellite orientation. In such case, the trajectory planning module takes the desired final position of the end-effector and the desired final orientation of the chaser satellite as an input. Information about the initial state of the system and obstacles is also required (we assume that the position and orientation of obstacles is constant in CS_{ine}). The trajectory planning module generates the reference manipulator trajectory (defined in the joint space) and the reference control torques for manipulator joints. It is the role of the trajectory following module to ensure that the manipulator joints are following this reference trajectory. This module uses measurements of the angular positions of manipulator joints in the feedback loop (more advanced controller than the one presented in Fig. 2 could also use measurements of joint velocities). Review of various approaches for control of the manipulator mounted on the chaser satellite can be found in ref. [55].

In the approach proposed in this article, the trajectory planning problem is solved with the RRT algorithm.[49] This algorithm is particularly tailored for problems defined in high-dimensional state spaces with first-order differential constraints and obstacle-based global constraints. The RRT algorithm is designed to efficiently search a state space by randomly building a tree that fills this space. It was successfully applied for fixed-base manipulators working on Earth.[56–58] The first application of the RRT algorithm for free-floating satellite-manipulator system was presented by Rybus and Seweryn[30] and, independently, by Benevides and Grassi.[31] The algorithm was used in refs. [30, 31] to find a collision-free trajectory, but the final orientation of the chaser satellite was not constrained. In this article, we propose to use the bi-directional RRT algorithm to find such a trajectory that will ensure the desired change of the satellite orientation.

Detailed description of the RRT algorithm can be found in ref. [49]. Our approach for application of the RRT algorithm for manipulator mounted on the free-floating satellite is based on ref. [30], but we introduce several improvements. We take advantage of the assumption that there are no external torques and forces acting on the system. In such case, the momentum and the angular momentum are conserved. Moreover, we assume that the momentum and the angular momentum are equal to zero. The centre of mass of the system is stationary: $\mathbf{r}_{CM} = const$ (we do not take into account the orbital motion as in the short time scale of the capture manoeuvre, this motion has no influence on the dynamics of the satellite-manipulator system). The state vector, defined in (2), has $12 + 2n$ com-

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

ponents, but in the considered case, only $6+2n$ components are independent. Using the equation that describes position of the system centre of mass, we can present $\mathbf{r}_s$ as a function of $\mathbf{r}_i$ $(\boldsymbol{\Theta}_s, \boldsymbol{\theta})$

$$\mathbf{r}_s = \mathbf{r}_{CM}\left(1 + \frac{1}{m_s}\sum_{i=1}^{n} m_i\right) - \frac{1}{m_s}\sum_{i=1}^{n} \mathbf{r}_i m_i \tag{18}$$

We differentiate (18) with respect to time, and we take into account the fact that the velocity of the centre of mass is always equal to zero ($\mathbf{v}_{CM}=0$). Now it is possible to present the linear velocity of the satellite as a function of velocity of the mass centres of manipulator links $\mathbf{v}_i\left(\boldsymbol{\Theta}_s, \boldsymbol{\omega}_s, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right)$:

$$\mathbf{v}_s = -\frac{1}{m_s}\sum_{i=1}^{n} \mathbf{v}_i m_i \tag{19}$$

Equations (18) and (19) allow us to use $6+2n$ dimensional state vector in the RRT algorithm:

$$\mathbf{x}_{RRT} = \begin{bmatrix} \boldsymbol{\omega}_s \\ \dot{\boldsymbol{\theta}} \\ \boldsymbol{\Theta}_s \\ \boldsymbol{\theta} \end{bmatrix} \tag{20}$$

In the initial state and in the final state, the end-effector position and the orientation of the satellite are specified. It is possible to compute $\boldsymbol{\theta}$ from $\mathbf{r}_{ee}$ and $\boldsymbol{\Theta}_s$ by solving the inverse kinematics problem (solution of this problem depends on the kinematics of the manipulator). We assume that the initial and final desired velocity of the end-effector is equal to zero, that is, $\mathbf{v}_{ee}(t=0)=0$, $\mathbf{v}_{ee}(t=t_f)=0$. In a consequence, the initial and the final angular velocity of the satellite is equal to zero: $\boldsymbol{\omega}_s(t=0)=0$ and $\boldsymbol{\omega}_s(t=t_f)=0$. Velocities of manipulator joints are also equal to zero: $\dot{\boldsymbol{\theta}}(t=0)=0$ and $\dot{\boldsymbol{\theta}}(t=t_f)=0$. Thus, state of the system is defined unambiguously. In ref. [30], the desired final orientation of the satellite was not specified. In such case, the desired final position of the end-effector can be reached for various orientations of the satellite (a given position of the end-effector corresponds to a curve in the $\boldsymbol{\Theta}_s\boldsymbol{\theta}$ space). Thus, it was not possible to use the bi-directional approach for the tree construction. Such approach for the tree construction is used in this article to find a collision-free trajectory that will ensure the desired change of the chaser satellite orientation. A detailed description of the bi-directional approach can be found in ref. [49]. In such approach, two trees are simultaneously constructed in the state space. One tree, “A”, is constructed from the initial state, while the second tree, “B”, is constructed backward in time from the final desired state. Pseudocode of the proposed bi-directional RRT algorithm applied for the free-floating satellite-manipulator system is presented in Algorithm 1. Work of the algorithm can be sketched as follows.

Initial state of the satellite-manipulator system, denoted as $\mathbf{x}_{t=0}$, is inserted as the first vertex of the “A” tree, while the final desired state, denoted as $\mathbf{x}_{t=t_f}$, is inserted as the first vertex of the “B” tree. The next steps of the algorithm are executed separately for each tree. A point ($\mathbf{x}_{goal}$) is selected at random in the state space, and a tree vertex nearest to this point is located. This vertex is denoted as $\mathbf{x}_{near}$ (in the beginning, it is the initial vertex of the tree). Alternatively, once every k_{bi} steps the point on the second tree that is nearest to the first tree is selected as $\mathbf{x}_{goal}$ instead of the randomly selected point. The selection of a tree vertex that is nearest to the given point depends on the metric used to measure the distance between two points in the state space. Various metrics can be used for that purpose.[59] We choose the following metrics based on a weighted Euclidean distance for the chaser satellite angular velocity, the velocity of manipulator joints, the orientation of the satellite and the positions of manipulator joints (these are the components of the state vector defined in (20)) to measure the distance between two points:

$$d = \zeta_1\|\Delta\boldsymbol{\omega}_s\| + \zeta_2\|\Delta\dot{\boldsymbol{\theta}}\| + \zeta_3\|\Delta\boldsymbol{\Theta}_s\| + \zeta_4\|\Delta\boldsymbol{\theta}\| \tag{21}$$

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

Algorithm 1 BUILD_BI-RRT()

```
Insert state $\mathbf{x}_{t=0}$ as the first vertex of $TreeA$;
Insert state $\mathbf{x}_{t=t_f}$ as the first vertex of $TreeB$;
for $i_{ver} \leftarrow 1$ to $n_{ver}$ do
    if $i_{ver}/k_{bi}$ is an integer then
        $\mathbf{x}_A$, $\mathbf{x}_B$ $\leftarrow$ Identify a pair of vertices (one vertex of $TreeA$ and one vertex of $TreeB$) for which the distance $d_{AB}$ is the smallest;
        $\mathbf{x}_{goal\_A} \leftarrow \mathbf{x}_B$;
        $\mathbf{x}_{near\_A} \leftarrow \mathbf{x}_A$;
        $\mathbf{x}_{goal\_B} \leftarrow \mathbf{x}_A$;
        $\mathbf{x}_{near\_B} \leftarrow \mathbf{x}_B$;
    else
        $\mathbf{x}_{goal\_A}$ $\leftarrow$ Randomly select point in the state-space;
        $\mathbf{x}_{near\_A}$ $\leftarrow$ Find the vertex of $TreeA$ for which the distance to $\mathbf{x}_{goal\_A}$ is the smallest;
        $\mathbf{x}_{goal\_B}$ $\leftarrow$ Randomly select point in the state-space;
        $\mathbf{x}_{near\_B}$ $\leftarrow$ Find the vertex of $TreeB$ for which the distance to $\mathbf{x}_{goal\_B}$ is the smallest;
    end if
    Extend $TreeA$ from $\mathbf{x}_{near\_A}$ towards $\mathbf{x}_{goal\_A}$;
    Extend $TreeB$ from $\mathbf{x}_{near\_B}$ towards $\mathbf{x}_{goal\_B}$;
end for
$\mathbf{x}_A$, $\mathbf{x}_B$ $\leftarrow$ Identify a pair of vertices (one vertex of $TreeA$ and one vertex of $TreeB$) for which the distance $d_{AB}$ is the smallest;
$T_{A\_temp}$ $\leftarrow$ Retrace trajectory from $\mathbf{x}_A$ to the first vertex of $TreeA$;
$T_A$ $\leftarrow$ Reverse trajectory $T_{A\_temp}$;
$T_B$ $\leftarrow$ Retrace trajectory from $\mathbf{x}_B$ to the first vertex of $TreeB$;
$T_{AB\_temp}$ $\leftarrow$ Connect $T_A$ and $T_B$ through $\mathbf{x}_A$ and $\mathbf{x}_B$ by shifting positions of manipulator joints in $T_B$;
$T_{AB}$ $\leftarrow$ Smooth the trajectory $T_{AB\_temp}$;
Perform the collision check for the smoothed trajectory $T_{AB}$;
$\mathbf{u}_{traj}$ $\leftarrow$ Compute control torques for $T_{AB}$;
```

Algorithm 2 EXTEND_TREE($Tree$, $\mathbf{x}_{near}$, $\mathbf{x}_{goal}$)

```
for $i \leftarrow 1$ to $i_u$ do
    $\mathbf{u}$ $\leftarrow$ Select the control torque from the predefined set;
    $\mathbf{x}_{candid}$ $\leftarrow$ Use RK IV method to compute state after application of $\mathbf{u}$ to $\mathbf{x}_{near}$;
    if $\mathbf{x}_{candid}$ does not violate constraints (joint limits, collisions) then
        $d_i$ $\leftarrow$ Compute distance between $\mathbf{x}_{candid}$ and $\mathbf{x}_{goal}$;
        if $i = 1 \vee d_i < d_{new}$ then
            $\mathbf{x}_{new} \leftarrow \mathbf{x}_{candid}$;
            $\mathbf{d}_{new} \leftarrow \mathbf{d}_i$;
        end if
    end if
end for
Add $\mathbf{x}_{new}$ as a new vertex of $Tree$ (with $\mathbf{x}_{near}$ as its parent);
```

where ζ_k denotes weighting coefficient, Δ denotes difference between the appropriate coordinates of two points and $\|z\|$ denotes the Euclidean norm of the vector: $\mathbf{z} = \left[z_1 \; \dots \; z_m\right]^T$:

$$\|z\| := \sqrt{z_1^2 + \dots + z_m^2} \tag{22}$$

For $\dot{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}$, we have $m = n$. In the three-dimensional case for $\boldsymbol{\omega}_s$ and $\boldsymbol{\Theta}_s$, we have $m = 3$, while in the planar case $m = 1$. Equation (21) is based on the equation presented in ref. [49]. The problem of selection of metric in the RRT algorithm is discussed in detail in ref. [59]. Different shapes of the trajectory tree will be obtained with a different combinations of weighting coefficients. We follow the approach presented in ref. [30]; and in computation of the distance between two points, we consider only selected components of the state vector (20). We assume $\zeta_1 = \zeta_2 = 0$. Thus, the distance depends only on the orientation of the satellite and positions of manipulator joints. In order to find the tree vertex that is nearest to the given point, the algorithm computes the distance between this point and each vertex of the tree.

The procedure EXTEND_TREE is used to find the control signal $\mathbf{u}$, which will bring the state from the vertex $\mathbf{x}_{near}$ towards the selected point $\mathbf{x}_{goal}$. Pseudocode of this procedure is presented in Algorithm 2. We are considering the free-floating system controlled by torques applied in the manipulator joints. Due to complicated dynamics of the satellite-manipulator system, finding a control signal $\mathbf{u}$ that will bring the system from a given state towards other selected state is a complex task. We are using a simple approach in which several combinations of control signals are considered. To obtain a new state of the satellite-manipulator system for a given control signal, the dynamic equations (17) are solved with the RK-IV method (in order to use Eq. (17), it is necessary to reconstruct the full state using Eqs. (18) and (19)). The new state is obtained after application of k_{RK} steps of this method. For $k_{RK}/2$ steps (we assume that k_{RK} is an even number) control torque in each joint can have only one of the two values: $u_i = \pm u_{RRT}$, where u_{RRT} is a constant value of the control torque (u_{RRT} may depend on the distance between $\mathbf{x}_{near}$ and $\mathbf{x}_{goal}$). For the subsequent $k_{RK}/2$ steps: $u_i = G_{RRT}\dot{\theta}_i$, where G_{RRT} is a constant gain. In such approach, velocities of manipulator joints in the new state are close to zero. For a given number of vertices, we obtain better coverage of the state space than in the case of constant torque applied for all k_{RK} steps (as in ref. [30]). In this approach, 2^n combinations of the control torques must be checked for the n-DoF manipulator; for example, for a 2-DoF manipulator, there are four possible combinations: (i) $u_1 = +u_{RRT}$, $u_2 = +u_{RRT}$, (ii) $u_1 = +u_{RRT}$, $u_2 = -u_{RRT}$, (iii) $u_1 = -u_{RRT}$, $u_2 = -u_{RRT}$ and (iv) $u_1 = -u_{RRT}$, $u_2 = +u_{RRT}$. For each combination, we obtain a different state of the system, denoted as $\mathbf{x}_{candid}$.

In the next step, the algorithm checks if, for any of these states, the constraints imposed on the motion planning problem are violated. If there is a state $\mathbf{x}_{candid}$ for which at least one of the constraints is violated, then the combination of the control torques that led to such state is rejected. In our approach, we take into account two types of constraints: (i) limits of positions of manipulator joints (arising from the mechanical construction of the joints) and (ii) collisions between the links of the manipulator and obstacles in the manipulator workspace.

The check for the first type of constraints is straightforward. For each joint of the manipulator, the algorithm tests if the following condition is fulfilled

$$(\theta_i)_{min} < \theta_i < (\theta_i)_{max} \tag{23}$$

where θ_i is the angular position of the i-th joint of the manipulator, while $(\theta_i)_{min}$ and $(\theta_i)_{max}$ denote the minimal and maximal allowed position of this joint.

The most likely obstacles that will be encountered during the ADR and OOS missions are the elements of the target satellites and, possibly, elements of large orbital structures that will be constructed or serviced with the autonomous satellite-manipulator systems. As explained in ref. [60], most satellites are composed of modular and standardised devices. The contours of these devices can be approximated by regular geometry shapes such as cuboid, cylinder or sphere. For the sake of simplicity, in our approach, we are only considering obstacles described as cuboids (in the three-dimensional case) and as rectangles (in the planar case). Each obstacle is defined by the position of its centre ($\mathbf{P}_O$), its orientation and the length of its edges.

Pseudocode of the COLLISION_CHECK procedure used to check for collisions between the links of the manipulator and obstacles in the manipulator workspace is presented in Algorithm 3. The collision check is performed separately for each obstacle. First, the position of cuboid vertices is calculated from $\mathbf{P}_O$, obstacle orientation and length of its edges. The vertices on one side of the obstacle are denoted by $\mathbf{r}_{O1}$, $\mathbf{r}_{O2}$, $\mathbf{r}_{O3}$, $\mathbf{r}_{O4}$ (numbered in the counter-clockwise direction), while vertices on the opposite side of the obstacle are denoted by $\mathbf{r}_{O5}$, $\mathbf{r}_{O6}$, $\mathbf{r}_{O7}$, $\mathbf{r}_{O8}$ (the pair $\mathbf{r}_{O1}$ and $\mathbf{r}_{O5}$ lies on one edge of the considered cuboid). The obstacle can be described by the following vectors: $\mathbf{v}_{12} = \mathbf{r}_{O2} - \mathbf{r}_{O1}$, $\mathbf{v}_{14} = \mathbf{r}_{O4} - \mathbf{r}_{O1}$ and $\mathbf{v}_{15} = \mathbf{r}_{O5} - \mathbf{r}_{O1}$.

Then, from the state vector $\mathbf{x}_{candid,}$ the position of the chaser satellite is computed using Eq. (18). After that, the forward kinematics problem is solved and position of the end-effector in CS_{ine} is calculated. Positions of the manipulator joints in the Cartesian space ($\mathbf{r}_{J1}, \mathbf{r}_{J2}, \ldots, \mathbf{r}_{Jn}$) are also computed.

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

Algorithm 3 COLLISION_CHECK($\mathbf{x}_{candid}$,$obstacle$)

```
$\mathbf{r}_{O1}, \mathbf{r}_{O2}, \cdots, \mathbf{r}_{O8} \leftarrow$ Calculate the positions of the obstacle vertices based on the position of the obstacle centre, obstacle orientation and its size;
$\mathbf{v}_{12} \leftarrow \mathbf{r}_{O2} - \mathbf{r}_{O1}$;
$\mathbf{v}_{14} \leftarrow \mathbf{r}_{O4} - \mathbf{r}_{O1}$;
$\mathbf{v}_{15} \leftarrow \mathbf{r}_{O5} - \mathbf{r}_{O1}$;
$\mathbf{r}_{J1}, \mathbf{r}_{J2}, \cdots, \mathbf{r}_{Jn}, \mathbf{r}_{ee} \leftarrow$ Solve forward kinematics problem for $\mathbf{x}_{candid}$ to compute positions of manipulator joints and position of the end-effector;
for $i \leftarrow 1$ to $n$ do
    $\mathbf{r}_{Lb} \leftarrow \mathbf{r}_{Ji}$;
    if $i < n$ then
        $\mathbf{r}_{Le} \leftarrow \mathbf{r}_{Ji+1}$;
    else
        $\mathbf{r}_{Le} \leftarrow \mathbf{r}_{ee}$;
    end if
    for $j \leftarrow 1$ to $h_{RRT}$ do
        $\mathbf{r}_{L} \leftarrow (\mathbf{r}_{Le} - \mathbf{r}_{Lb}) \cdot j/h_{RRT} + \mathbf{r}_{Lb}$;
        $\mathbf{v}_{1L} \leftarrow \mathbf{r}_{L} - \mathbf{r}_{O1}$;
        if $0 < \mathbf{v}_{1L} \cdot \mathbf{v}_{12} < \mathbf{v}_{12} \cdot \mathbf{v}_{12} \wedge 0 < \mathbf{v}_{1L} \cdot \mathbf{v}_{14} < \mathbf{v}_{14} \cdot \mathbf{v}_{14} \wedge \ldots$
          $0 < \mathbf{v}_{1L} \cdot \mathbf{v}_{15} < \mathbf{v}_{15} \cdot \mathbf{v}_{15}$ then
            Collision detected, break the loop;
        end if
    end for
end for
```

The solution of the forward kinematics problem for fixed-base manipulators can be found, for example, in ref. [61]. In the case of free-floating space manipulator, we have to take into account also the position and orientation of the chaser satellite (with respect to CS_{ine}).

The next part of the COLLISION_CHECK procedure is repeated for each link of the manipulator. In Algorithm 3, one end of the link is denoted by $\mathbf{r}_{Lb}$, while the other end is denoted by $\mathbf{r}_{Le}$. Positions of the endpoints coincide with $\mathbf{r}_{J1}, \mathbf{r}_{J2}, \ldots \mathbf{r}_{Jn}$. The only exception is the last link: position of one end of this link coincides with the position of the manipulator end-effector ($\mathbf{r}_{ee}$). The link is divided into h_{RRT} equal sections (h_{RRT} should be selected taking into account dimensions of obstacles and length of manipulator links). Positions of $h_{RRT} - 1$ intermediate points (given in CS_{ine}) are calculated using the following equation:

$$\mathbf{r}_L = (\mathbf{r}_{Le} - \mathbf{r}_{Lb})\frac{j}{h_{RRT}} + \mathbf{r}_{Lb} \tag{24}$$

where $\mathbf{r}_L$ is the position of the j-th point on the considered link. This point lies within the cuboid if all of the following conditions are fulfilled

$$\begin{aligned} 0 &< \mathbf{v}_{1L} \cdot \mathbf{v}_{12} < \mathbf{v}_{12} \cdot \mathbf{v}_{12} \\ 0 &< \mathbf{v}_{1L} \cdot \mathbf{v}_{14} < \mathbf{v}_{14} \cdot \mathbf{v}_{14} \\ 0 &< \mathbf{v}_{1L} \cdot \mathbf{v}_{15} < \mathbf{v}_{15} \cdot \mathbf{v}_{15} \end{aligned} \tag{25}$$

where $\mathbf{v}_{1L} = \mathbf{r}_L - \mathbf{r}_{O1}$. When the first point that lies within the obstacle is encountered, then there is no need to check other points. In such case, the state $\mathbf{x}_{candid}$ is rejected. For each state that does not lead to constraints violation, the algorithm calculates the distance to the selected point. The state that is closest to this point is inserted as a new vertex of the tree ($\mathbf{x}_{new}$) with $\mathbf{x}_{near}$ as its parent.

If the defined number of tree vertices, n_{ver}, is not yet reached, the algorithm returns to the selection of a new point (randomly placed in the state space or located on the second tree). When n_{ver} is reached, two vertices (one on the “A” tree and another on the “B” tree) that are closest to each other are selected. These vertices are denoted as $\mathbf{x}_A$ and $\mathbf{x}_B$, respectively. Equation (21) is used to calculate the distance between points (distance between the selected pair, $\mathbf{x}_A$ and $\mathbf{x}_B$, is denoted as d_{AB}). Each vertex contains information about its parent, so it is possible to backtrack the trajectory from $\mathbf{x}_A$

Table I. Basic parameters of the chaser satellite.

Parameter	**Value**
Mass	60 kg
Mass moment of inertia	1.875 kg·m^2
External dimensions of the satellite (X, Y)	0.5 m × 0.5 m
Position of the first manipulator joint in respect to the mass centre (X, Y)	(0.4 m, 0 m)

Table II. Basic parameters of the 2-DoF manipulator attached to the satellite.

Parameter	**Link 1**	**Link 2**
Length	0.6 m	0.6 m
Mass	4.5 kg	1.5 kg
Position of the centre of mass (along the link)	0.3 m	0.3 m
Mass moment of inertia	0.135 kg·m^2	0.045 kg·m^2
Range of allowable positions of manipulator joints	$(-150°, +150°)$	$(-170°, +150°)$

and $\mathbf{x}_B$ to the first vertex of each tree (each point of the trajectory defines the state of the satellite-manipulator system). The trajectory obtained from the "A" tree must be reversed, as the last point on this trajectory coincides with the initial state of the system (the first vertex of the "A" tree). The last point on the trajectory obtained from the "B" tree coincides with the final desired state of the satellite-manipulator system. These two trajectories, T_A and T_B, are connected together to form the solution of the trajectory planning problem.

As pointed out in ref. [62], connecting two parts of the trajectory obtained from the bi-directional algorithm is especially difficult in case of nonholonomic systems (free-floating satellite equipped with a manipulator is such a system). As explained in ref. [49] several different techniques can be used to ensure continuity of the obtained trajectory, for example, classical shooting techniques. In our approach, we connect these two trajectories in the joint space. The positions of manipulator joints for the trajectory T_B are shifted in such a way that the first point of this trajectory in the joint space coincides exactly with the last point of the trajectory T_A. After that, the first point of the trajectory T_B does not coincide exactly with the state in the vertex $\mathbf{x}_B$, but the continuity of the resulting trajectory T_{AB} is ensured. However, after realisation of the trajectory T_{AB} , the desired end-effector position and the desired chaser satellite orientation will be obtained with some error. As will be shown in Section 7, this error depends on the distance d_{AB} between $\mathbf{x}_A$ and $\mathbf{x}_B$.

The smoothing of the trajectory T_{AB} in the joint space is done using a simple moving average. For the final trajectory, the collision check is performed in order to make sure that changes to the original trajectory do not result in collisions with the obstacles. In the last stage, Eq (13) is used to compute new control torques for the entire trajectory T_{AB}. At this stage, time of motion can be arbitrarily selected, because the velocity with which the manipulator moves on the defined trajectory does not influence the final orientation of the chaser satellite. In practical applications, due to possible disturbances, control system must be used to ensure realisation of this trajectory.[51]

5. Results of Numerical Simulations

To verify the proposed approach, we performed numerical simulations for a simplified case. We consider a planar free-floating satellite equipped with a 2-DoF manipulator in which both joints are rotational. The parameters of this system are summarised in Tables I and II (the selected parameters are not representative for a real space mission but for a mock-up of a satellite-manipulator system that can be used in experiments on a planar air-bearing microgravity simulator). In the considered case, the satellite-manipulator system has a high ratio of the mass of the manipulator to the mass of the satellite. Control of such system is challenging, because influence of the manipulator motion on the state of the satellite is significant (a high-dynamic coupling exists between the manipulator and the satellite). The similar planar system, but with different parameters, was considered in ref. [48], where polynomials were used to obtain the desired final state of the system. Thus, selection of a planar case allows comparison of our approach with previously presented approach.

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

The CS_{ine} is located in the initial position of the satellite mass centre. The initial position of the end-effector, given in CS_{ine}, is $\mathbf{r}_{ee}(t=0)=\left[0.7\,\text{m}\ -0.1\,\text{m}\right]^T$. At $t=0$, the components of the state vector of the satellite-manipulator system ($\mathbf{x}=\left[\mathbf{v}_s^T\ \boldsymbol{\omega}_s^T\ \dot{\boldsymbol{\theta}}^T\ \mathbf{r}_s^T\ \boldsymbol{\Theta}_s^T\ \boldsymbol{\theta}^T\right]^T$) have the following values: $\mathbf{v}_s(t=0)=\left[(v_s)_x\ (v_s)_y\right]^T=\left[0\ 0\right]^T$, $\boldsymbol{\omega}_s(t=0)=(\omega_s)_z=0$, $\dot{\boldsymbol{\theta}}(t=0)=\left[\dot{\theta}_1\ \dot{\theta}_2\right]^T=\left[0\ 0\right]^T$, $\mathbf{r}_s(t=0)=\left[(r_s)_x\ (r_s)_y\right]^T=\left[0\ 0\right]^T$, $\boldsymbol{\Theta}_s(t=0)=\psi_s=0$, $\boldsymbol{\theta}(t=0)=\left[\theta_1\ \theta_2\right]^T=\left[0.982\ -2.608\right]^T$. There is one square obstacle in the workspace. The centre of this obstacle is located in the $\mathbf{P}_O=\left[0.895\,\text{m}\ 0.025\,\text{m}\right]^T$. The side of the obstacle has the length $a=0.1$ m. Sides of the obstacle are parallel to the axes of CS_{ine}. The final desired position of the end-effector is $\mathbf{r}_T=\mathbf{r}_{ee}(t=t_f)=\left[1.2\,\text{m}\ -0.1\,\text{m}\right]^T$, while the desired final orientation of the satellite is $\psi_s(t=t_f)=0.349=20°$.

To find the collision-free trajectory that ensures the desired change of the chaser satellite orientation, two trees are constructed. The initial state of the system ($\mathbf{x}_{t=0}$) is inserted as the first vertex of the "A" tree, while the desired final state ($\mathbf{x}_{t=t_f}$) is inserted as the first vertex of the "B" tree (constructed backwards in time). Each tree is constructed from $n_{ver}=50{,}000$ vertices. During construction of these trees, new vertices are added after $k_{RK}=10$ steps of the RK-IV method with the time step $t_{RK}=0.01$ s. The nearest point on the second tree is selected as $\mathbf{x}_{goal}$ once every $k_{bi}=2$ executions of the algorithm's main loop. Control torque used during the tree construction is $u_{RRT}=0.5$ Nm, while the control gain is $G_{RRT}=-5$. During the collision check, each link of the manipulator is divided into $h_{RRT}=10$ equal sections.

The results obtained with the proposed approach are compared with results obtained with two other methods: (i) the method based on the basic RRT algorithm (one-directional)[30] and (ii) the APF method.[63] These two methods allow to find a collision-free trajectory of the manipulator from a given initial state to a final state in which the end-effector is in the desired position. However, they do not allow to obtain the desired final orientation of the chaser satellite. As explained in the Introduction, no other methods are known that allow to obtain these two goals at the same time. The purpose of the presented comparison is to show how the final orientation of the chaser satellite obtained with the basic RRT algorithm and the APF method will differ from the desired value.

In the basic RRT algorithm (one-directional), only one tree is built from the initial state of the system.[49] After construction of the tree, the vertex for which the position of the end-effector is closest to the desired final end-effector position is selected. The trajectory is then constructed (using the vertices of the tree) from the initial state to this selected tree vertex. For construction of this single tree, we use the same approach as the one presented in Section 4 (the only difference is that in every step, the tree is extended towards the randomly selected state). The same set of parameters was used in the bi-directional approach, but the tree was constructed from $n_{ver}=65{,}000$ vertices.

In the APF method, the manipulator is treated as a particle that moves under the influence of an artificial force field.[64] A scalar function (called the potential) is constructed in such a way that it has its minimum in the desired final configuration of the manipulator and has high values in the configurations with obstacles. We follow the approach presented in ref. [63], where the potential field is constructed in the physical space. The attracting force acts on the end-effector in the direction of the desired final end-effector position, while the repulsive force generated by obstacles acts on the links of the manipulator. We use exactly the same parameters as presented in ref. [63].

The "A" tree obtained from the RRT method is shown in Fig. 3 in the space of manipulator joints positions, while in Fig. 4, this tree is shown in the space of the end-effector position. The "B" tree is shown in Figs. 5 and 6. The positions of manipulator joints during realisation of the trajectory are shown in Fig. 7, where the results obtained from the proposed method based on the bi-directional RRT algorithm are compared with the results obtained from the method based on the basic RRT algorithm (one-directional) and the APF method. The end-effector position obtained from each method is shown in Fig. 8. Trajectories of the end-effector on the XY plane are presented on the left panel of Fig. 9, while the orientation of the chaser satellite is shown on the right panel of Fig. 9. The satellite-manipulator system during realisation of the trajectory obtained from the proposed method based on the bi-directional RRT algorithm is presented in Fig. 10. Velocities of manipulator joints during realisation of this trajectory are presented in Fig. 11. Control torques for the entire trajectory are shown in Fig. 12, while Fig. 13 presents these torques in two selected time periods: from $t=4s$ to $t=5s$

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

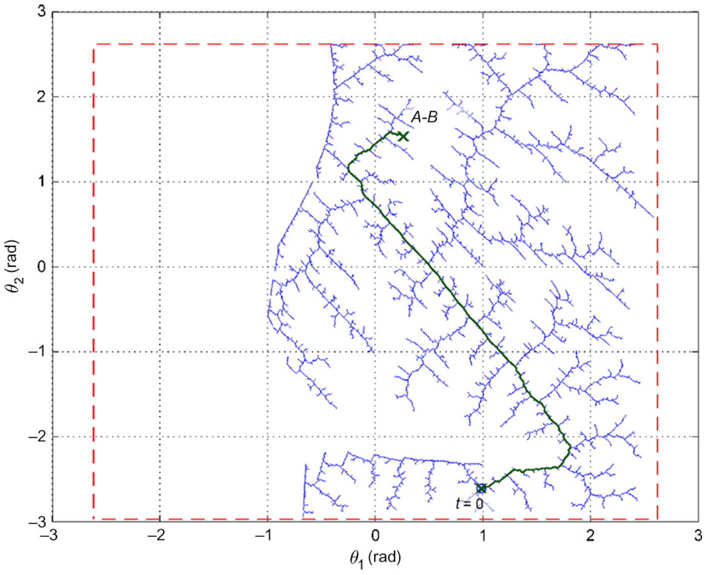

Fig. 3. The “A” tree in the space of manipulator joints positions. The manipulator trajectory from the initial state to the vertex that connects the “A” tree with the “B” tree is marked in green. The limits of manipulator joints are marked as dashed red lines.

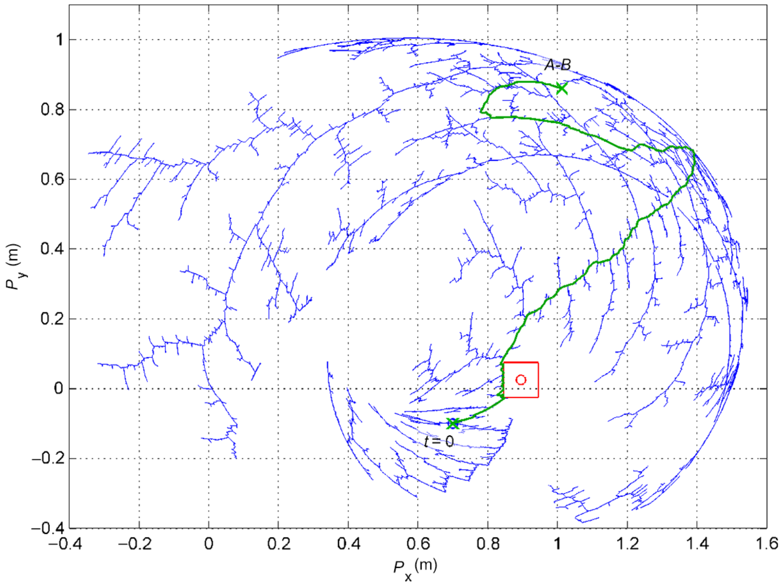

Fig. 4. The “A” tree in the space of the end-effector position. The manipulator end-effector trajectory from the initial end-effector position to the position that corresponds to the vertex that connects the “A” tree with the “B” tree is marked in green. The obstacle is marked as a red square.

and from $t = 10.5$ s to $t = 11.5$ s. During the first period, the end-effector moves in close proximity to the obstacle, while the second period covers the connection point between T_A and T_B (this point is at $t_{AB} = 11.15$ s). Obtained results are summarised in Table III. The error in the final end-effector position is defined as

$$\Delta \mathbf{r}_{ee} = \sqrt{[(r_{ee\_sim})_x - (r_T)_x]^2 + [(r_{ee\_sim})_y - (r_T)_y]^2} \tag{26}$$

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

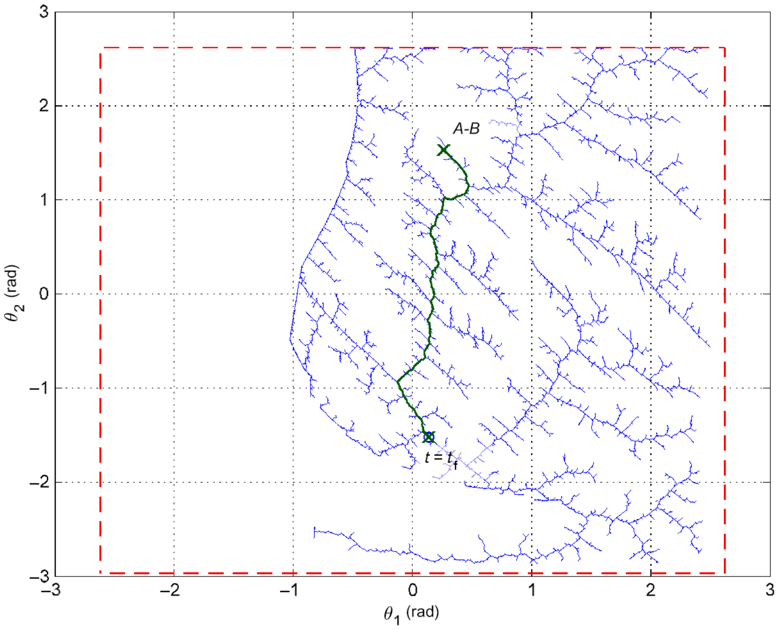

Fig. 5. The “B” tree in the space of manipulator joints positions. The manipulator trajectory from the vertex that connects the “B” tree with the “A” tree to the final desired state is marked in green. The limits of manipulator joints are marked as dashed red lines.

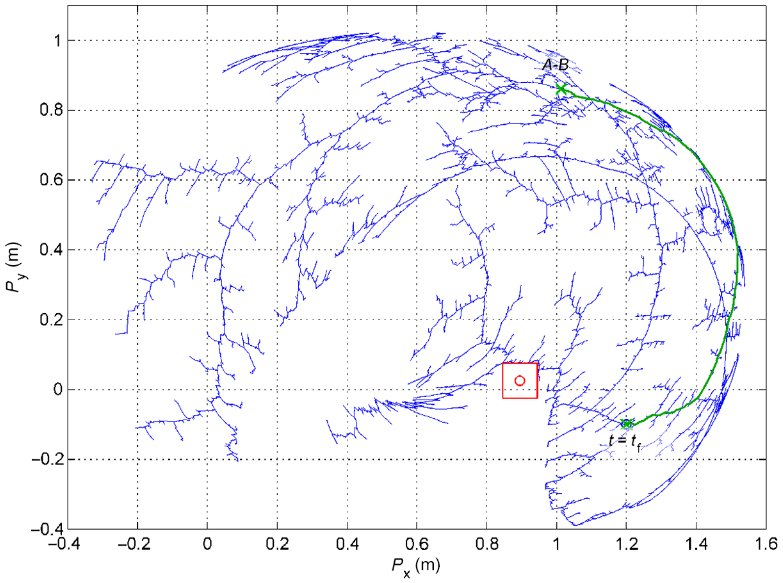

Fig. 6. The “B” tree in the space of the end-effector position. The manipulator end-effector trajectory from the position that corresponds to the vertex that connects the “B” tree with the “A” tree to the final desired end-effector position is marked in green. The obstacle is marked as a red square.

where $\mathbf{r}_{ee\_sim} = \left[(r_{ee\_sim})_x \ (r_{ee\_sim})_y\right]^T$ is the final end-effector position obtained from numerical simulation for the given trajectory defined in the joint space, while $(r_T)_x$ and $(r_T)_y$ are the components of the desired final end-effector position. Error in the final orientation of the chaser satellite is defined as

$$\Delta\psi_s = |(\psi_s)_{sim} - \psi_s(t = t_f)| \tag{27}$$

where $(\psi_s)_{sim}$ is the final orientation of the chaser satellite obtained from numerical simulation.

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

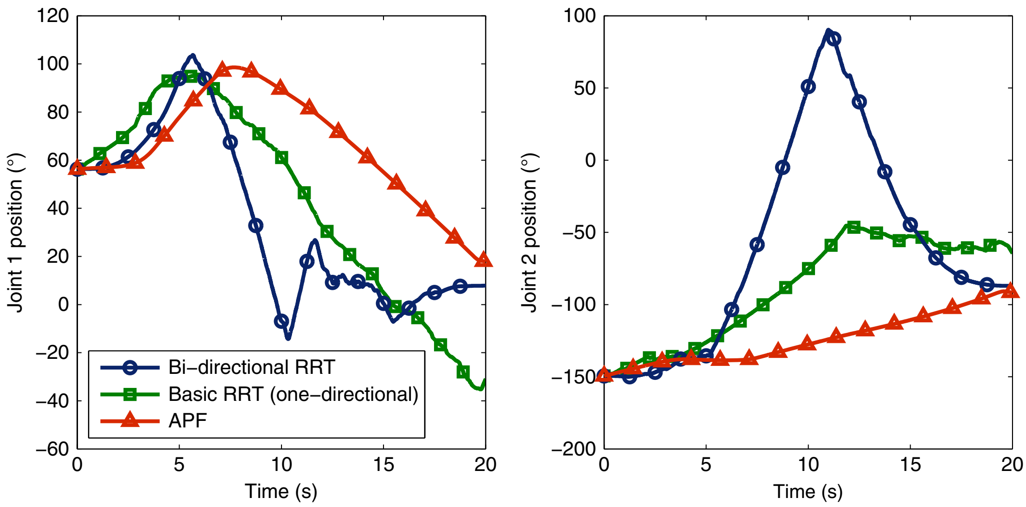

Fig. 7. The position of the manipulator joint 1 (left panel) and joint 2 (right panel) during realisation of the trajectory obtained from (i) the proposed method based on the bi-directional RRT algorithm, (ii) the method based on the simple RRT algorithm (one-directional) and (iii) the APF method.

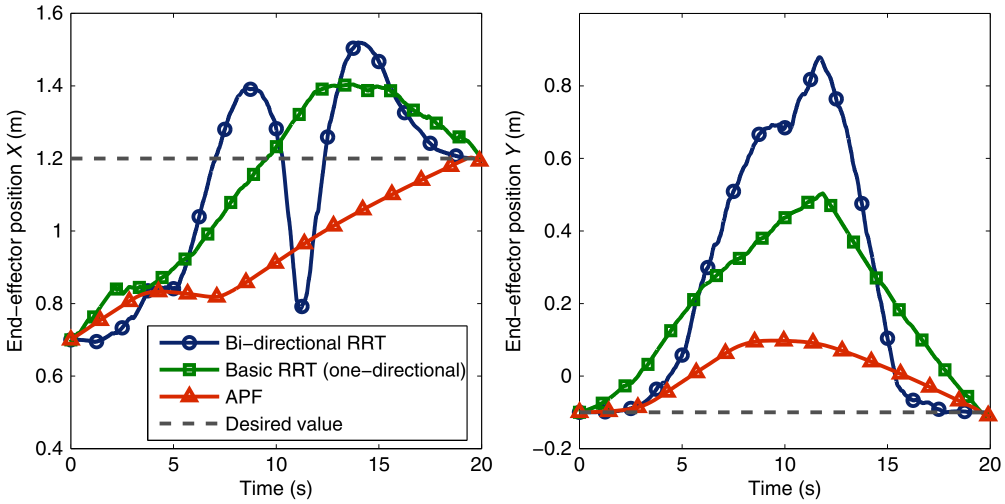

Fig. 8. The X-component (left panel) and the Y-component (right panel) of the manipulator end-effector position during realisation of the trajectory obtained from (i) the proposed method based on the bi-directional RRT algorithm, (ii) the method based on the simple RRT algorithm (one-directional) and (iii) the APF method.

6. Discussion

The bi-directional RRT algorithm constructed the “A” tree starting from the initial state of the satellite-manipulator system and the “B” tree starting from the desired final state. To find a collision-free trajectory from the initial state to the final state (in which the chaser satellite obtains the desired final orientation and the manipulator end-effector reaches the desired final position) these two trees are connected. Two vertices ($\mathbf{x}_A$ on the “A” tree and $\mathbf{x}_B$ on the “B” tree) that are closest to each other are selected.

Figures 3 and 5 were constructed by projecting the six-dimensional trees into two-dimensional space of manipulator joints positions. The connection point between the “A” tree and “B” tree is denoted as “A-B” (the actual distance between $\mathbf{x}_A$ and $\mathbf{x}_B$ is equal to d_{AB}). These two trees could be presented together as a single tree on one plot, but in such case, the projections of tree branches would overlap. Figures 3 and 5 show that the space, in which the tree is constructed, is uniformly covered with vertices. However, areas, in which there are no vertices, are clearly visible. These areas correspond to the states of the satellite-manipulator system, in which the manipulator collides with

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

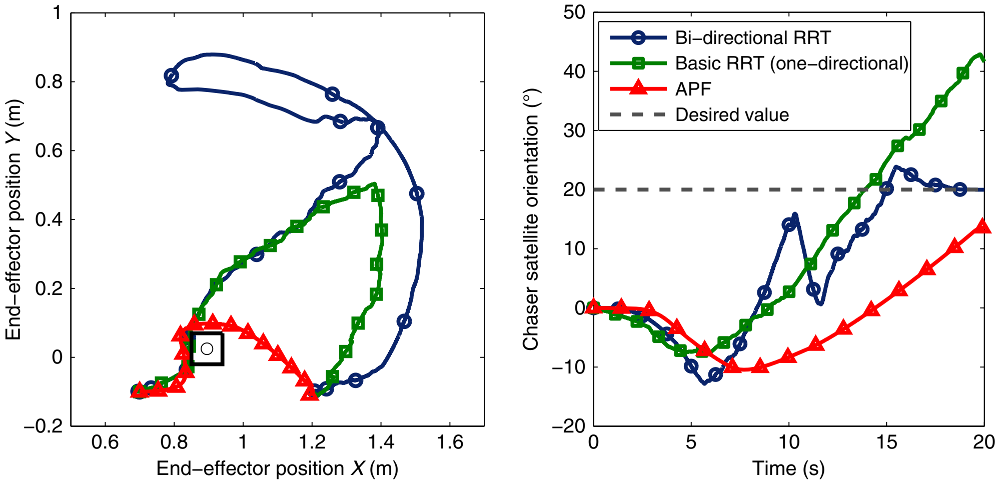

Fig. 9. The end-effector trajectory on the *XY* plane (left panel) and orientation of the chaser satellite (right panel) during realisation of the trajectory obtained from (i) the proposed method based on the bi-directional RRT algorithm, (ii) the method based on the simple RRT algorithm (one-directional) and (iii) the APF method.

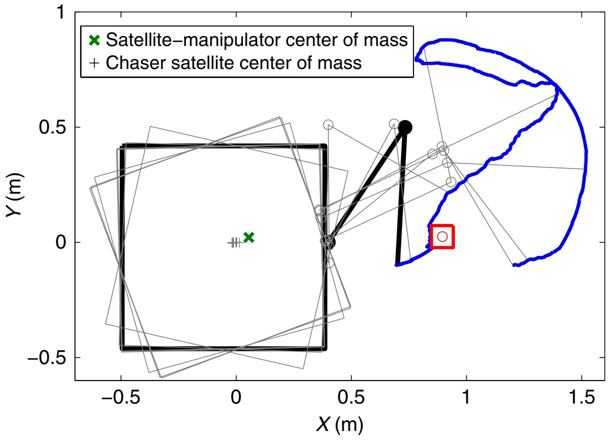
Satellite−manipulator center of mass
Chaser satellite center of mass
Y (m)
X (m)
1
0.5
0
−0.5
−0.5
0
0.5
1
1.5

Fig. 10. The satellite-manipulator system during realisation of the trajectory obtained from the proposed method based on the bi-directional RRT algorithm.

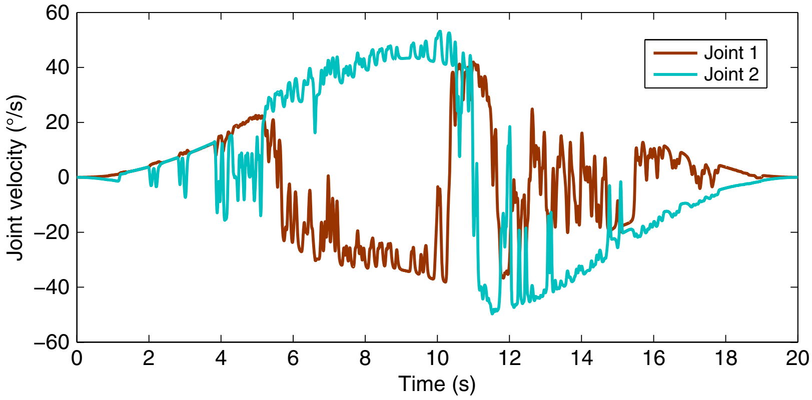

Fig. 11. Velocities of manipulator joints obtained from the proposed method based on the bi-directional RRT algorithm.

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

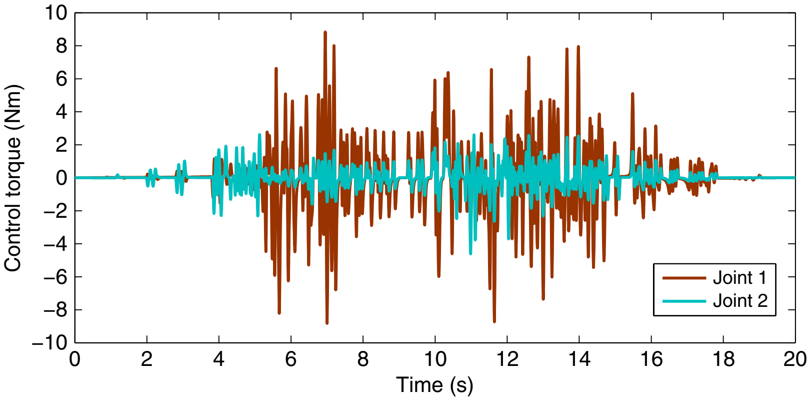

Fig. 12. Control torques in the manipulator joints obtained from the proposed method based on the bi-directional RRT algorithm.

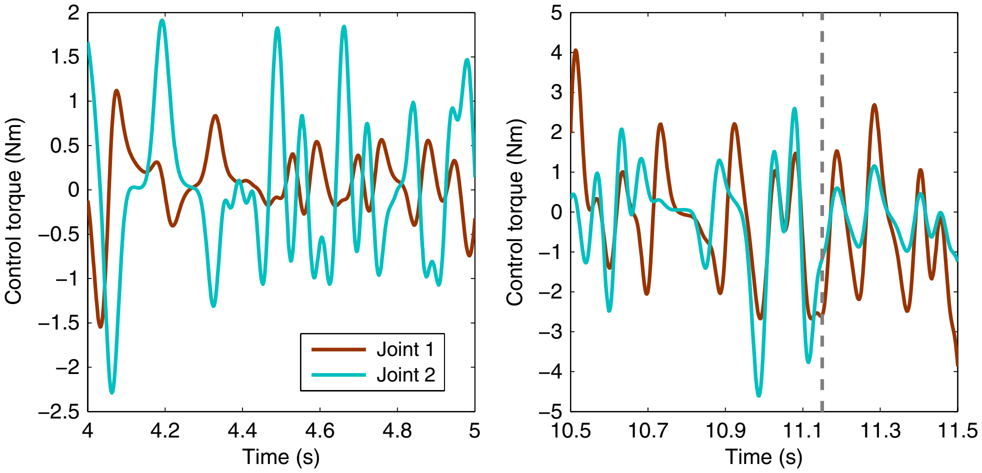

Fig. 13. Control torques in the manipulator joints obtained from the proposed method based on the bi-directional RRT algorithm: time period from $t = 4$ s to $t = 5$ s (left panel) and from $t = 10.5$ s to $t = 11.5$ s (right panel). The connection point between T_A and T_B is marked as a dashed grey line.

the obstacle. Parts of the space cannot be reached due to the limited range of the allowable positions of manipulator joints.

Figures 4 and 6 were constructed by projecting the trees into two-dimensional space of the end-effector position. It can be seen that on each of these figures, the projections of some branches of the tree overlap. This is because the same end-effector position is reached for various orientations of the satellite. One square area in Figs. 4 and 6 is not covered with vertices, because the obstacle is in this area. Other areas are not covered with vertices because of the limited range of the manipulator.

The manipulator trajectory T_{AB} (combined from the "A" tree and "B" tree) is shown in Fig. 7 in the joint space. The continuity of this trajectory in the joint space was ensured by shifting the positions of manipulator joints for the trajectory T_B in such a way that the first point of this trajectory in the joint space coincides exactly with the last point of the trajectory T_A. The positions of the manipulator joints change from $\boldsymbol{\theta}(t = 0) = \begin{bmatrix}0.982 & -2.608\end{bmatrix}^T$ to $\boldsymbol{\theta}(t = t_f) = \begin{bmatrix}0.137 & -1.521\end{bmatrix}^T$. As can be seen from Fig. 8, the selected manipulator trajectory resulted in the motion of the manipulator end-effector to the desired final position $\mathbf{r}_{ee}(t = t_f) = \begin{bmatrix}1.2\text{ m} & -0.1\text{ m}\end{bmatrix}^T$. In Figs. 7 and 8, the results obtained from the proposed method based on the bi-directional RRT algorithm are compared with the results obtained from the method based on the simple RRT algorithm (one-directional) and the APF method. Each method resulted in a very different trajectory. However, in all three methods, the final end-effector position obtained from the numerical simulations is very close to the desired position $\mathbf{r}_T$ (the highest end-effector position error was obtained for the APF method).

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

Table III. Final end-effector position and chaser satellite orientation obtained with different trajectory planning methods.

Parameter	Bi-directional RRT	Basic RRT (one-directional)	APF
End-effector position X (m)	1.2006	1.1997	1.1902
End-effector position Y (m)	−0.0999	−0.0998	−0.1114
End-effector position error (m)	0.00064	0.00034	0.01503
Chaser satellite orientation (°)	19.9831	41.6377	13.6292
Chaser satellite orientation error (°)	0.0169	21.6377	6.3708

In this study, we are considering a free-floating satellite-manipulator system. Thus, reaction forces and reaction torques induced by the motion of the manipulator influence position and orientation of the satellite. This effect is clearly seen in Fig. 10. During realisation of the manipulator trajectory obtained from the bi-directional RRT algorithm, the chaser satellite changes its orientation and its position (however, the mass centre of the entire satellite-manipulator system remains in a fixed position). As shown in Fig. 9, at the end of the manipulator trajectory obtained with the proposed approach, the chaser satellite reaches the desired orientation: $\psi_s(t = t_f) = 20°$ (error in the chaser satellite orientation is smaller than 0.017°). The final orientation of the chaser satellite obtained after realisation of trajectories found with the method based on the basic RRT algorithm and the APF method is very different from the desired value (for the method based on the basic RRT algorithm $(\psi_s)_{sim} = 41.64°$, while for the APF method: $(\psi_s)_{sim} = 13.63°$). As explained in Section 5, these two other methods used for comparison were only ensuring collision-free motion of the manipulator end-effector to the desired final position. The comparison presented in Fig. 9 and Table III shows the advantage of the proposed approach for trajectory planning.

Velocities of manipulator joints and control torques calculated for the smoothed trajectory obtained from the proposed method based on the bi-directional RRT algorithm are presented in Figs. 11 and 12, respectively. The maximal control torques are below 9 Nm. Possibility to realise this trajectory on a real system (e.g. using a mock-up of the satellite-manipulator system operated on a planar air-bearing microgravity simulator) depends on the capabilities of such system. However, results shown in Figs. 12 and 13 suggest that for systems such as the one presented in ref. [65] the obtained trajectory is feasible and could be realised in experiments. Moreover, as explained in Section 4, the velocity with which the manipulator moves on the defined trajectory does not influence the final orientation of the chaser satellite. Thus, if the considered scenario allows longer time of motion, then such time can be selected to reduce the maximal values of the control torques.

7. Analysis of the Trajectory Planning Method

In this section, detailed analysis of the proposed trajectory planning method is presented. The trajectory planning for the case considered in Section 5 was repeated several times for different numbers of tree vertices n_{ver}. In the proposed algorithm, the trees are being constructed until the desired number of vertices is reached. However, a pair of vertices, $\mathbf{x}_A$ and $\mathbf{x}_B$ that are sufficiently close to connect the two trees, can be found before the given number of vertices is reached. After such pair is found, it is possible that the addition of new vertices will not generate a pair for which the distance d_{AB} (calculated using (21)) will be smaller. Let us denote by n_{AB} the number of vertices for which the closest distance between the two trees is obtained. This number is presented on the left panel of Fig. 14 as a function of d_{AB}. The results are presented for 22 different cases with $n_{ver} = \{10{,}000, 25{,}000, 50{,}000, 75{,}000\}$. It should be noted that, by definition, $n_{AB} \leq n_{ver}$. It can be seen that smaller values of d_{AB} are usually obtained for trees with higher n_{ver} (points representing the results of trajectory planning performed with the higher number of tree vertices are located on the left-hand side of this plot). In some cases, the pair of vertices, $\mathbf{x}_A$ and $\mathbf{x}_B$, for which the distance between the two trees is the smallest, was obtained on an early stage of the tree construction, that is, for low value of n_{AB}. The addition of new vertices does not guarantee that the pair of points will be obtained, for which d_{AB} would be smaller.

On the right panel of Fig. 14, difference between the desired and obtained orientation of the chaser satellite is presented as a function of the difference between the desired and obtained position of the end-effector. This plot shows the influence of n_{ver} on the final error. This error results only from the

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

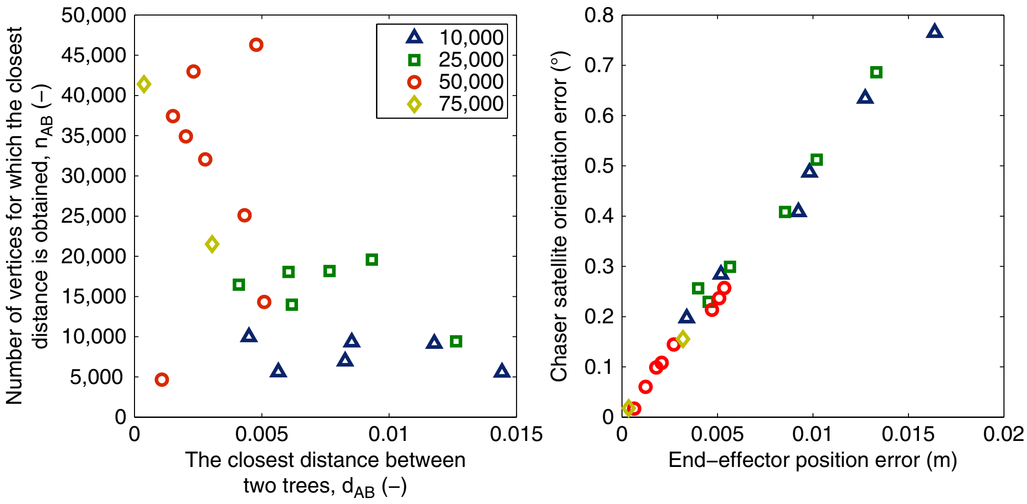

Fig. 14. Number of vertices for which the closest distance between the two trees is obtained as a function of this distance (left panel) and difference between the desired and obtained orientation of the chaser satellite as a function of the difference between the desired and obtained position of the end-effector (right panel) for trees constructed with $n_{ver} = \{10{,}000, 25{,}000, 50{,}000, 75{,}000\}$.

fact that the trajectory T_B obtained from the "B" tree had to be shifted in order to ensure continuity in the joint space of the resulting trajectory T_{AB}. Thus, because the first point of the trajectory T_B does not coincide exactly with the state in the vertex $\mathbf{x}_B$, the desired end-effector position and the desired chaser satellite orientation are obtained with some error. The increase in the number of tree vertices could allow the algorithm to find the solution, for which d_{AB} is smaller and, as a result, the end-effector position error and the chaser satellite orientation error at the end of the trajectory are also smaller. To further investigate the influence of the distance between $\mathbf{x}_A$ and $\mathbf{x}_B$ on the final errors, these errors are presented in Fig. 15 as a function of d_{AB}. Which value of d_{AB} can be treated as sufficiently small depends on which errors are acceptable. The presented plots allow to judge what value of d_{AB} can be treated as sufficiently small.

Results for different cases with $n_{ver} = \{10{,}000, 25{,}000, 50{,}000, 75{,}000\}$ are summarised in Table IV. As expected, the average values of the final end-effector position error and the chaser satellite orientation error, obtained for all cases with the same value of n_{ver}, are smaller for trees constructed with higher number of vertices. Higher value of n_{ver} increases the chance of finding a pair of vertices, $\mathbf{x}_A$ and $\mathbf{x}_B$, for which d_{AB} is smaller, and, as a result, errors are also smaller. However, the trees are constructed randomly and, in some cases, better results are obtained for a tree constructed with a smaller number of vertices (e.g. for one case with $n_{ver} = 50{,}000$, we obtained $d_{AB} = 0.0051$ and the final chaser satellite orientation error of 0.257°, while for one case with $n_{ver} = 25{,}000$, we obtained $d_{AB} = 0.0041$ and the final chaser satellite orientation error of 0.23°).

If the distance d_{AB} is not small enough, then the obtained trajectory T_{AB} may not be collision-free. This results from the fact that positions of manipulator joints for the trajectory T_B are shifted to ensure continuity of T_{AB} in the joint space. Thus, the positions of manipulator links in the Cartesian space may differ from the collision-free positions obtained for the original trajectory T_B. However, this problem was not observed in any of the analysed cases. The minimal number of tree vertices, $n_{ver} = 10{,}000$, was sufficient to obtain sufficiently small value of d_{AB} and the resulting trajectory T_{AB} that is collision-free.

Results presented in Section 5 cover only one scenario, that is, one set of initial conditions and one desired end-effector position and orientation of the chaser satellite. As shown above, for this condition, the proposed method based on the bi-directional RRT algorithm was able to find a solution in every run. However, one of the major drawbacks of the proposed approach (and some other methods based on the RRT algorithm) is that it is not possible to define a set of conditions for which it is guaranteed that the algorithm will find a solution. Problem of defining such set is common for methods devoted to the collision-free trajectory planning of free-floating satellite-manipulator systems (it is difficult to determine if any solution exists for the considered case).[23] In the next part of this section, we will investigate the influence of two selected parameters (chaser satellite mass and inertia) on the

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

Table IV. The summary of results obtained for 22 runs of the proposed trajectory planning method based on the bi-directional RRT algorithm.

Number of tree vertices		**10,000**	**25,000**	**50,000**	**75,000**
Number of vertices for which d_{AB} is obtained	Min	5576	9429	4657	21,524
	Max	19,985	19,585	46,305	41,413
	Average	7770	15,950	29,721	31,468
The closest distance between the two trees, d_{AB}	Min	0.0045	0.0041	0.0011	0.0004
	Max	0.0144	0.0126	0.0051	0.0030
	Average	0.0089	0.0077	0.0030	0.0017
Chaser satellite orientation error (°)	Min	0.1973	0.2295	0.0169	0.0179
	Max	0.7653	0.6863	0.2570	0.1552
	Average	0.4628	0.3986	0.1419	0.0866
End-effector position error (m)	Min	0.0034	0.0040	0.0006	0.0004
	Max	0.0164	0.0133	0.0054	0.0032
	Average	0.0095	0.0077	0.0030	0.0018

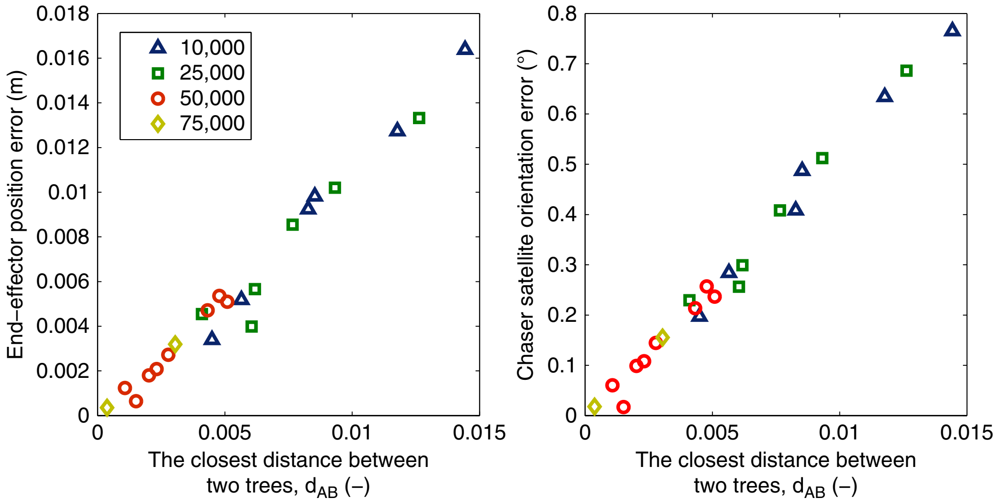

Fig. 15. The end-effector position error (left panel) and the chaser satellite orientation error (right panel) as a function of the smallest distance between the two trees for trees constructed with $n_{ver} = \{10{,}000, 25{,}000, 50{,}000, 75{,}000\}$.

performance of the proposed trajectory planning method. The trajectory planning was performed for the scenario defined in Section 5 with $n_{ver} = 25{,}000$. Results obtained for the nominal values of the chaser satellite mass and inertia (given in Table I) were compared with results obtained for the 50% lower and 50% higher mass and inertia. The positions of manipulator joints during realisation of the trajectory are shown in Fig. 16, while the end-effector position is shown in Fig. 17. Trajectories of the end-effector on the *XY* plane are presented on the left panel of Fig. 18, while the orientation of the chaser satellite is shown on the right panel of Fig. 18. Obtained results are summarised in Table V.

The proposed method based on the bi-directional RRT algorithm was able to find a collision-free trajectory that ensures desired final orientation of the chaser satellite for the case with 50% reduced mass and inertia and for the case with 50% increased mass and inertia of the chaser satellite. As can be seen from Fig. 16, in each of the three analysed cases, the same final positions of manipulator joints were obtained. The final desired state of the satellite-manipulator system ($\mathbf{x}_{t=t_f}$) is defined by the desired final position of the end-effector and final orientation of the chaser satellite. As we are considering planar 2-DoF manipulator, there are only two sets of manipulator joints positions that for the given orientation of the chaser satellite result also in the desired position of the manipulator end-effector. The final end-effector position error and the chaser satellite orientation error obtained for the case with reduced mass and inertia are smaller than errors obtained for the case with nominal

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

Table V. Final end-effector position and chaser satellite orientation obtained with the proposed method based on the bi-directional RRT algorithm for different mass and inertia of the chaser satellite: (i) $0.5 \cdot m_s$, $0.5 \cdot I_s$ (50% reduced mass and inertia), (ii) m_s, I_s (nominal mass and inertia) and (iii) $1.5 \cdot m_s$, $1.5 \cdot I_s$ (50% increased mass and inertia).

Parameter	**Reduced mass and inertia**	**Nominal mass and inertia**	**Increased mass and inertia**
End-effector position X (m)	1.1997	1.1988	1.2022
End-effector position Y (m)	−0.1010	−0.1085	−0.0852
End-effector position error (m)	0.0011	0.0085	0.0150
Chaser satellite orientation (°)	19.9606	19.5918	20.6522
Chaser satellite orientation error (°)	0.0394	0.4082	0.6522

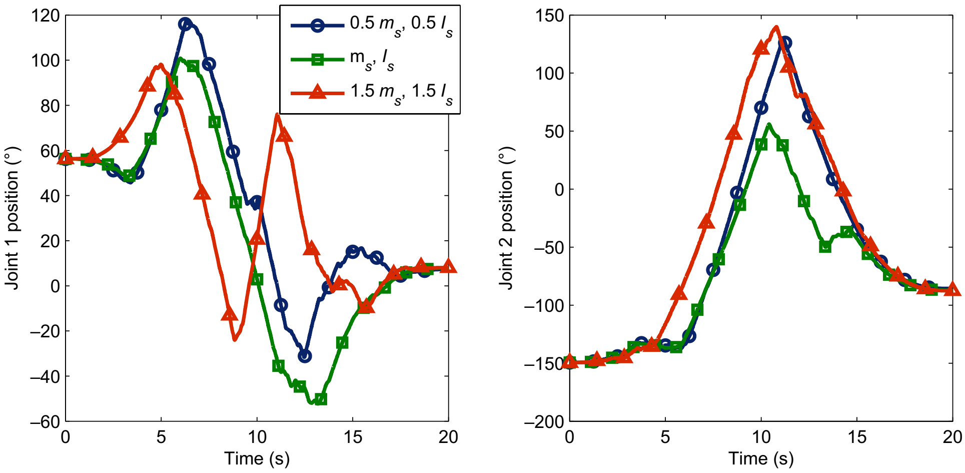

Fig. 16. The position of the manipulator joint 1 (left panel) and joint 2 (right panel) during realisation of the trajectory obtained from the proposed method based on the bi-directional RRT algorithm for three different values of the chaser satellite mass and inertia: (i) $(m_s)_{red} = 30\,\text{kg}$, $(I_s)_{red} = 0.9375\,\text{kg}\cdot\text{m}^2$, (ii) $m_s = 60\,\text{kg}$, $I_s = 1.875\,\text{kg}\cdot\text{m}^2$ and (iii) $(m_s)_{inc} = 90\,\text{kg}$, $(I_s)_{inc} = 2.8125\,\text{kg}\cdot\text{m}^2$

parameters. They are also smaller than the average errors obtained for all runs of the proposed method with nominal parameters and $n_{ver} = 25{,}000$ (presented in Table IV). The highest values of errors were obtained for the increased mass and inertia of the chaser satellite. This results from the fact, that for the increased mass and inertia, the influence of the manipulator motion on the state of the chaser satellite is smaller. Thus, more significant motions of the manipulator are required to obtain desired rotation of the chaser satellite. For the increased mass and inertia of the chaser satellite, higher number of vertices is required to obtain value of d_{AB} similar to the value obtained for nominal parameters.

Although in the example presented in Section 5, a simplified planar case is considered, the dynamic equations presented in Section 3 and the trajectory planning method based on the bi-directional RRT algorithm (shown in Section 4) can be applied in the three-dimensional case. The main difference would be in the number of components of the state vector used by the RRT algorithm (for the planar case, this number is equal to $2 + 2n$, while for the three-dimensional case, it is equal to $6 + 2n$). The difference in the number of components of this vector results from the fact that, in the planar case, only one number is needed to describe the orientation of the chaser satellite, while, in the three-dimensional case, three numbers are needed. The higher dimension of the state space would complicate the construction of the tree. However, the proposed approach could still be used.

The proposed approach could also be applied for system with a redundant manipulator (equations presented in Sections 3 and 4 are valid for any n-DoF manipulator). The only difference between the use of the nonredundant and redundant manipulator is the length of the vector of generalised coordinates $\mathbf{q}_p$ and the length of the state vector in the RRT algorithm $\mathbf{x}_{RRT}$. In case of the planar example presented in Section 5, use of a redundant 3-DoF manipulator would result in increase of the length of the state vector $\mathbf{x}_{RRT}$ from 6 to 8. Thus, the trees would be built in the eight-dimensional

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

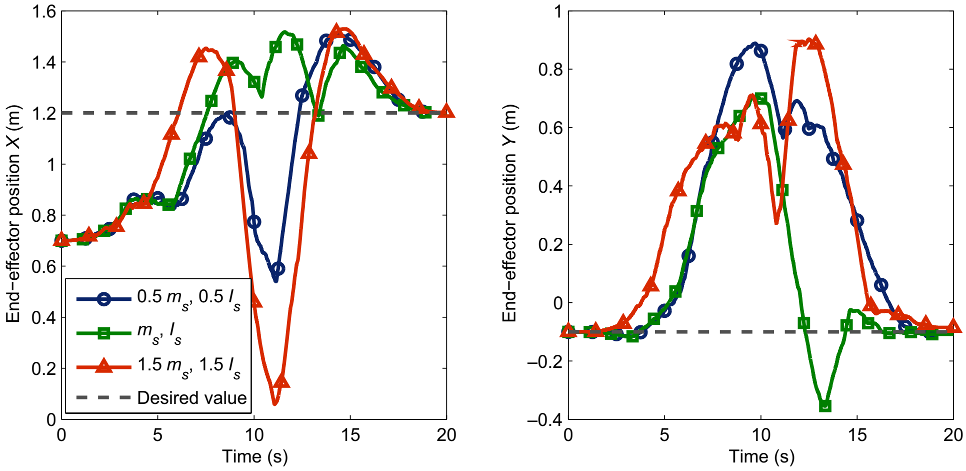

Fig. 17. The X-component (left panel) and the Y-component (right panel) of the manipulator end-effector position during realisation of the trajectory obtained from the proposed method based on the bi-directional RRT algorithm for three different values of the chaser satellite mass and inertia: (i) $(m_s)_{red} = 30$ kg, $(I_s)_{red} = 0.9375$ kg·m^2, (ii) $m_s = 60$ kg, $I_s = 1.875$ kg·m^2 and (iii) $(m_s)_{inc} = 90$ kg, $(I_s)_{inc} = 2.8125$ kg·m^2

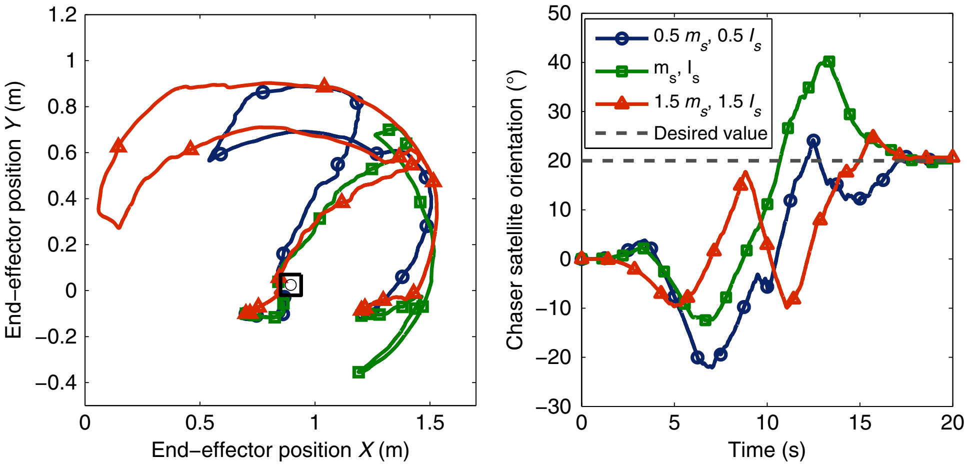

Fig. 18. The end-effector trajectory on the XY plane (left panel) and orientation of the chaser satellite (right panel) during realisation of the trajectory obtained from the proposed method based on the bi-directional RRT algorithm for three different values of the chaser satellite mass and inertia: (i) $(m_s)_{red} = 30$ kg, $(I_s)_{red} = 0.9375$ kg·m^2, (ii) $m_s = 60$ kg, $I_s = 1.875$ kg·m^2 and (iii) $(m_s)_{inc} = 90$ kg, $(I_s)_{inc} = 2.8125$ kg·m^2

space. However, case with the redundant manipulator is less challenging for the trajectory planning algorithm. Moreover, it seems that in such case other methods (e.g. the APF method) could also allow to plan the collision-free trajectory and, at the same time, to obtain the desired final orientation of the satellite. This results from the fact that, for the given end-effector motion, the additional DoFs of the manipulator could be used to obtain the desired change of the chaser satellite state. As mentioned in the Introduction, the first algorithm that allows such simultaneous control of the redundant manipulator end-effector and orientation of the satellite was proposed in ref. [38], while experiments with a planar free-floating satellite equipped with a 3-DoF manipulator were presented in ref. [39]. No collisions were considered in both cases, but the idea to use the additional DoFs of the redundant manipulator to control the orientation of the chaser satellite can be incorporated into the collision-free trajectory planning methods.

In this article, the satellite-manipulator system with zero linear and angular momentum was considered. In such case, the centre of mass of the system is stationary. The approach presented in Section 4 takes advantage of this simplifying assumption: the number of components in the state vector used by the RRT algorithm is reduced from $12 + 2n$ to $6 + 2n$. As a result, construction of the tree in the state space is simplified. However, as explained in ref. [30], the basic RRT algorithm could also be used for collision-free trajectory planning in systems with non-zero momentum and angular momentum. The general equations of the satellite-manipulator dynamics, presented in Section 3, can be extended for the systems with non-zero and even non-conserved linear and angular momentum.[51,65] However, in such case, Eqs. (18) and (19) will not be valid and the full $12 + 2n$ dimensional state vector must be used for construction of the tree. Nevertheless, the general trajectory planning methodology based on the bi-directional RRT algorithm should be applicable for systems with non-zero and non-conserved linear and angular momentum.

In the near future, it is planned to test the proposed trajectory planning method in more complicated cases, for example, the three-dimensional case with a redundant manipulator (such case would be much closer to the real conditions that will be encountered during the planned ADR and OOS missions). The next important step will be the optimisation of the presented algorithm and its implementation in C++. After this, it will be possible to evaluate the computational time required for the trajectory planning. These steps are needed to assess the possibility of using the proposed approach in ADR and OOS missions. It seems that the high computational complexity of the proposed method may be its major drawback. The high computational cost of the trajectory planning may be especially problematic in case of the algorithm implementation on the flight hardware. However, the trajectory planning stage can be performed, while the satellite-manipulator system is waiting in a safe point. It might even be possible to perform such computations on Earth.

8. Conclusions

We have presented application of the RRT algorithm for trajectory planning of the nonredundant manipulator mounted on the free-floating satellite. The bi-directional approach is used in the construction of the tree in order to plan a trajectory from the given initial state to the specific final state. The proposed approach allows planning a collision-free manipulator trajectory that, at the same time, will result in the desired change of the chaser satellite orientation. Moreover, several improvements are introduced in comparison to the previous application of the RRT method for the free-floating satellite-manipulator system. These improvements include the reduction of the dimension of the state space to simplify construction of the tree and use of a new approach for selection of control signals during construction of the tree to obtain better coverage of the state space. The proposed approach for trajectory planning was verified in numerical simulations performed for a simplified planar case in which the chaser satellite is equipped with the 2-DoF manipulator. Presented results of these numerical simulations proved that the RRT algorithm can be successfully used to solve the obstacle avoidance problem and to provide the desired change of the chaser satellite orientation. Two other methods used for comparison (the method based on the basic RRT algorithm and the APF method) do not allow to obtain these two goals simultaneously. The proposed approach has a practical significance in the context of the planned OOS and ADR missions. In these missions, the manipulator will operate in close proximity to obstacles (such as the elements of the target satellite). Changes of the chaser satellite attitude, caused by the motions of the manipulator, could disturb communication link or power generation. However, further analysis, including the estimation of computational cost, is required in order to thoroughly evaluate the applicability of the presented method in a real space mission.

Acknowledgements

This article was partially supported by the Polish National Science Center project no. 2015/17/B/ST7/03995.

References

1. D. E. Hastings, B. L. Putbrese and P. A. La Tour, "When will on-orbit servicing be part of the space enterprise?" *Acta Astronaut.* **127**, 655–666 (2016).
2. A. R. Graham and J. Kingston, "Assessment of the commercial viability of selected options for on-orbit servicing (OOS)," *Acta Astronaut.* **117**, 38–48 (2015).
3. J. N. Pelton, *New Solutions for the Space Debris Problem* (Springer, Heidelberg, 2015).
4. J.-C. Liou, "An active debris removal parametric study for LEO environment remediation," *Adv. Space Res.* **47**(11), 1865–1876 (2011).
5. R. Biesbroek, L. Innocenti, A. Wolahan and S. M. Serrano, "e. Deorbit - ESA's Active Debris Removal Mission," *Proceedings of the 7th European Conference on Space Debris*, Darmstadt, Germany (2017).

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

6. M. A. Skinner, “Orbital debris: what are the best near-term actions to take? A view from the field,” *J. Space Saf. Eng.* **4**(2), 105–111 (2017).
7. G. Visentin and D. L. Brown, “Robotics for geostationary satellite servicing,” *Rob. Auton. Syst.* **23**(1), 45–51 (1998).
8. D. A. Smith, C. Martin, M. Kassebom, H. Petersen, A. Shaw, B. Skidmore, D. Smith, H. Stokes and A. Willig, “A mission to preserve the geostationary region,” *Adv. Space Res.* **34**(5), 1214–1218 (2004).
9. G. Hausmann, M. Wieser, R. Haarmann, A. Brito and J. C. Meyer, “e.Deorbit Mission: OHB Debris Removal Concepts,” *Proceedings of the 13th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2015)*, ESTEC, Noordwijk, The Netherlands (2015).
10. R. Zappulla, J. Virgili-Llop, C. Zagaris, H. Park and M. Romano, “Dynamic air-bearing hardware-in-the-loop testbed to experimentally evaluate autonomous spacecraft proximity maneuvers,” *J. Spacecraft Rockets* **54**(4), 825–839 (2017).
11. I. Rekleitis, E. Martin, G. Rouleau, R. L’Archeveque, K. Parsa and E. Dupuis, “Autonomous capture of a tumbling satellite,” *J. Field Robot.* **24**(4), 275–296 (2007).
12. C. R. Carignan and D. L. Akin, “The reaction stabilization of on-orbit robots,” *IEEE Control Syst. Mag.* **20**(6), 19–33 (2000).
13. M. Oda and Y. Ohkami, “Coordinated control of spacecraft attitude and space manipulators,” *Control Eng. Pract.* **5**(1), 11–21 (1997).
14. O. Egeland and J. R. Sagli, “Coordination of motion in a spacecraft/manipulator system,” *Int. J. Rob. Res.* **12**(4), 366–379 (1993).
15. E. Papadopoulos and S. Dubowsky, “Coordinated Manipulator/Spacecraft Motion Control for Space Robotic Systems,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1991)*, Sacramento, CA, USA (1991).
16. S. Jaekel, R. Lampariello, W. Rackl, B. Brunner, O. Porges, E. Kraemer, M. Pietras, J. Ratti, and R. Biesbroek, “Robotic Aspects and Analyses in the Scope of the e.Deorbit Mission Phase B1,” *Proceedings of the 14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2017)*, Leiden, The Netherlands (2017).
17. J. Telaar, I. Ahrns, S. Estable, W. Rackl, M. De Stefano, R. Lampariello, N. Santos, P. Serra, M. Canetri, F. Ankersen and J. Gil-Fernandez, “GNC Architecture for the e. Deorbit Mission,” *Proceedings of the 7th European Conference for Aeronautics and Space Sciences (EUCASS 2017)*, Milan, Italy (2017).
18. R. Rank, Q. Mühlbauer, W. Naumann and K. Landzettel, “The DEOS Automation and Robotics Payload,” *Proceedings of the 11th ESA Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA 2011)*, ESTEC, Noordwijk, The Netherlands (2011).
19. T. Rybus and K. Seweryn, “Manipulator Trajectories During Orbital Servicing Mission: Numerical Simulations and Experiments on Microgravity Simulator,” *Proceedings of the 6th European Conference for Aeronautics and Space Sciences (EUCASS 2015)*, Kraków, Poland (2015).
20. S. Dubowsky and E. Papadopoulos, “The kinematics, dynamics, and control of free-flying and free-floating space robotic systems,” *IEEE Trans. Robot. Autom.* **9**(5), 531–543 (1993).
21. A. Ellery, “An engineering approach to the dynamic control of space robotic on-orbit servicers,” *Proc. Inst. Mech. Eng. Part G: J. Aerosp. Eng.* **218**(2), 79–98 (2004).
22. Y. K. Hwang and N. Ahuja, “Gross motion planning–a survey,” *ACM Comput. Surv.* **24**(3), 219–291 (1992).
23. T. Rybus, “Obstacle avoidance in space robotics: Review of major challenges and proposed solutions,” *Prog. Aerosp. Sci.* **101**, 31–48 (2018).
24. R. Lampariello, “On grasping a tumbling debris object with a free-flying robot,” *IFAC Proc. Volumes* **46**(19), 161–166 (2013).
25. C. Toglia, M. Sabatini, P. Gasbarri and G. B. Palmerini, “Optimal target grasping of a flexible space manipulator for a class of objectives,” *Acta Astronaut.* **68**(7), 1031–1041 (2011).
26. R. Mukherjee and Y. Nakamura, “Nonholonomic Redundancy of Space Robots and its Utilization via Hierarchical Lyapunov Functions,” *Proceedings of the American Control Conference (ACC 1991)*, Boston, MA, USA (1991).
27. Y. Yanoshita and S. Tsuda, “Space Robot Path Planning for Collision Avoidance,” *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS 2009)*, vol. 2, Hong Kong (2009).
28. T. V. Bhargava and K. K. Issac, “Minimum time collision-free trajectories for grabbing a non-tumbling satellite,” *IFAC-PapersOnLine* **49**(1), 142–147 (2016).
29. X. Gao, Q. Jia, H. Sun and G. Chen, “Research on path planning for 7-DOF space manipulator to avoid obstacle based on A* algorithm,” *Sens. Lett.* **9**(4), 1515–1519 (2011).
30. T. Rybus and K. Seweryn, “Application of Rapidly-Exploring Random Trees (RRT) Algorithm for Trajectory Planning of Free-Floating Space Manipulator,” *Proceedings of the 10th International Workshop on Robot Motion and Control (RoMoCo 2015)*, Poznań, Poland (2015).
31. J. R. Benevides and V. Grassi, “Autonomous Path Planning of Free-Floating Manipulators Using RRT-Based Algorithms,” *Proceedings of the 12th Latin American Robotics Symposium and 3rd Brazilian Symposium on Robotics (LARS-SBR)*, Uberlandia, Minas Gerais, Brazil (2015).
32. J. B. Balaram and H. W. Stone, “Automated Assembly in the JPL Telerobot Testbed,” **In**: *Intelligent Robotic Systems for Space Exploration* (A. A. Desrochers, ed.) (Springer, New York, 1992) pp. 297–342.
33. R. K. Mathur, R. Münger and A. C. Sanderson, “Hierarchical Planning for Space-Truss Assembly,” **In**: *Intelligent Robotic Systems for Space Exploration* (A. A. Desrochers, ed.) (Springer, New York, 1992) pp. 141–184.

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

34. P. Huang and Y. Xu, "Attitude Compensation of Space Robots for Capturing Operation," **In**: *Mobile Robots: Towards New Applications* (A. Lazinica, ed.) (InTech, 2006) pp. 499–512.
35. M. Oda, "Coordinated Control of Spacecraft Attitude and its Manipulator," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1996)*, Minneapolis, Minnesota, USA (1996).
36. H. Sone and D. N. Nenchev, "On Some Practical Reactionless Motion Tasks with a Free-Floating Space Robot," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2015)*, Seattle, WA, USA (2015).
37. Z. Vafa and S. Dubowsky, "On the Dynamics of Manipulators in Space Using the Virtual Manipulator Approach," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1987)*, Raleigh, NC, USA (1987).
38. Z. Vafa, "Space Manipulator Motions with No Satellite Attitude Disturbances," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1990)*, Cincinnati, OH, USA (1990).
39. M. Marchesi, F. Angrilli and R. Venezia, "Coordinated Control for Free-Flyer Space Robots," *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, Nashville, TN, USA (2000).
40. F. Zhang, Y. Fu, L. Hua, H. Chen, S. Wang and B. Guo, "Point-to-Point Planning for Free-Floating Space Manipulator with Zero-Disturbance Spacecraft Attitude," *Proceedings of the IEEE International Conference on Information and Automation (ICIA 2012)*, Seoul, Korea (2012).
41. P. Piersigilli, I. Sharf and A. K. Misra, "Reactionless capture of a satellite by a two degree-of-freedom manipulator," *Acta Astronaut.* **66**(1–2), 183–192 (2010).
42. K. Yoshida, K. Hashizume and S. Abiko, "Zero Reaction Maneuver: Flight Validation with ETS-VII Space Robot and Extension to Kinematically Redundant Arm," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2001)*, Seoul, Korea (2001).
43. Y. Nakamura and R. Mukherjee, "Nonholonomic path planning of space robots via a bidirectional approach," *IEEE Trans. Robot. Autom.* **7**(4), 500–514 (1991).
44. C. Fernandes, L. Gurvits and Z. X. Li, "Attitude Control of Space Platform/Manipulator System Using Internal Motion," **In**: *Space Robotics: Dynamics and Control* (Y. Xu and T. Kanade, eds.) (Springer, Boston, 1993) pp. 131–163.
45. W. Xu, C. Li, X. Wang, Y. Liu, B. Liang and Y. Xu, "Study on non-holonomic cartesian path planning of a free-floating space robotic system," *Adv. Robotics* **23**(1–2), 113–143 (2009).
46. E. Papadopoulos, I. Tortopidis and K. Nanos, "Smooth Planning for Free-Floating Space Robots Using Polynomials," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2005)*, Barcelona, Spain (2005).
47. I. Tortopidis and E. Papadopoulos, "Point-to-Point Planning: Methodologies for Underactuated Space Robots," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2006)*, Orlando, FL, USA (2006).
48. I. Tortopidis and E. Papadopoulos, "On point-to-point motion planning for underactuated space manipulator systems," *Rob. Auton. Syst.* **55**(2), 122–131 (2007).
49. S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.* **20**(5), 378–400 (2001).
50. K. Seweryn and M. Banaszkiewicz, "Optimization of the Trajectory of a General Free-Flying Manipulator During the Rendezvous Maneuver," *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit (AIAA-GNC 2008)*, Honolulu, Hawaii, USA (2008).
51. T. Rybus, K. Seweryn and J. Z. Sasiadek, "Control system for free-floating space manipulator based on nonlinear model predictive control (NMPC)," *J. Intell. Robot. Syst.* **85**(3), 491–509 (2017).
52. Y. Umetani and K. Yoshida, "Resolved motion rate control of space manipulators with generalized Jacobian matrix," *IEEE Trans. Robot. Autom.* **5**, 303–314 (1989).
53. J. L. Junkins and H. Schaub, "An instantaneous eigenstructure quasivelocity formulation for nonlinear multibody dynamics," *J. Astronaut. Sci.* **45**(3), 279–295 (1997).
54. J. Kindracki, K. Tur, P. Paszkiewicz, L. Mezyk, L. Boruc and P. Wolanski, "Experimental research on low-cost cold gas propulsion for a space robot platform," *Aerosp. Sci. Technol.* **62**, 148–157 (2017).
55. A. Flores-Abad, O. Ma, K. Pham and S. Ulrich, "A review of space robotics technologies for on-orbit servicing," *Prog. Aerosp. Sci.* **68**, 1–26 (2014).
56. M. V. Weghe, D. Ferguson and S. S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," *Proceedings of the 7th IEEE-RAS International Conference on Humanoid Robots*, Pittsburgh, USA (2007).
57. A. Sintov and A. Shapiro, "Time-Based RRT Algorithm for Rendezvous Planning of Two Dynamic Systems," *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA 2014)*, Hong Kong, China (2014).
58. D. Berenson, S. S. Srinivasa, D. Ferguson and J. J. Kuffner, "Manipulation Planning on Constraint Manifolds," *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA 2009)*, Kobe, Japan (2009).
59. S. LaValle and J. Kuffner, "Rapidly-Exploring Random Trees: Progress and Prospects," **In**: *Mobile Robots: Towards New Applications* (B. R. Donald, K. M. Lynch and D. Rus, eds.) (A. K. Peters, Wellesley, MA, 2001) pp. 293–308.
60. Z. Mu, W. Xu, X. Gao, L. Xue and C. Li, "Obstacles Modeling and Collision Detection of Space Robots for Performing On-Orbit Services," *Proceedings of the 4th IEEE International Conference on Information Science and Technology (ICIST'2014)*, Shenzhen, China (2014).

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press

61. K. J. Waldron and J. Schmiedeler, “Kinematics,” **In**: *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.) (Springer, New York, 2016) pp. 11–36.
62. I. Noreen, A. Khan and Z. Habib, “Optimal path planning using RRT* based approaches: a survey and future directions,” *Int. J. Adv. Comput. Sci. Appl.* **7**, 97–107 (2016).
63. T. Rybus and K. Seweryn, “Zastosowanie metody sztucznych pól potencjału do planowania trajektorii manipulatora satelitarnego,” **In**: *Postepy Robotyki*, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa (K. Tchon and C. Zielinski, eds.) (2018) pp. 61–74 [in polish: Application of the artificial potential field method for trajectory planning of space manipulator].
64. O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” *Int. J. Rob. Res.* **5**(1), 396–404 (1986).
65. T. Rybus, K. Seweryn, J. Oles, F. L. Basmadji, K. Tarenko, R. Moczydlowski, T. Barcinski, J. Kindracki, L. Mezyk, P. Paszkiewicz and P. Wolanski, “Application of a planar air-bearing microgravity simulator for demonstration of operations required for an orbital capture with a manipulator,” *Acta Astronaut.* **155**, 221–229 (2019).

https://doi.org/10.1017/S0263574719001176 Published online by Cambridge University Press