IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 16, 2019, accepted December 12, 2019, date of publication December 20, 2019,
date of current version January 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2961167

Obstacle Avoidance and Path Planning for Multi-Joint Manipulator in a Space Robot

YAEN XIE[1], ZHIDAN ZHANG[2], XIANDE WU[2], ZHEN SHI[1], YANGYANG CHEN[3], BAIXUAN WU[2], AND KOFI AKROFI MANTEY[2]

[1]College of Automation, Harbin Engineering University, Harbin 150001, China
[2]College of Aerospace and Civil Engineering, Harbin Engineering University, Harbin 150001, China
[3]Engineering Specialist, Beijing Institute of Electronic System Engineering, Beijing 100854, China

Corresponding author: Zhen Shi (shizhen@hrbeu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 11772185, in part by the Fundamental Research Funds for the Central Universities under Grant HEUCFP201770, in part by the Natural Science Foundation of Heilongjiang Province under Grant F2017005), in part by the project under D020214, in part by the Shanghai Science and Technology Commission under Grant 16XD1421000, in part by the Joint Laboratory of Integrated Dynamics and Control for Spacecraft, and in part by the Joint Laboratory of Space Guidance and Control Technology.

ABSTRACT An obstacle avoidance and path planning algorithm for a multi-joint manipulator in a space robot is presented in this paper. In this paper, the end-effector of the manipulator is used to capture some special target in a space environment with obstacles. To ensure the safety of the operation, a collision-free path from the initial position to the target position is essential. Therefore, an obstacle avoidance and path planning algorithm based on the Rapidly-Exploring Random Tree (RRT) algorithm and the Forward and Backward Reaching Inverse Kinematics (FABRIK) algorithm is presented in this paper. First, a path planning algorithm based on the Rapidly-Exploring Random Tree (RRT) algorithm is designed for the multi-joint manipulator. Further, a method to generate a random point by artificial guidance is introduced for a higher searching speed. The RRT algorithm can effectively explore the entire workspace and find a feasible path without collision for the end-effector. To calculate the positions of each joint, the Forward and Backward Reaching Inverse Kinematics (FABRIK) algorithm is introduced and improved for the problem of inverse kinematics. The FABRIK algorithm avoids the use of rotational angles or matrices, and instead finds each joint position by locating a point on a line, and thus, it has a low computational cost. Therefore, the improved obstacle avoidance and path planning algorithm can quickly plan a feasible path for the multi-joint manipulator in a space environment with obstacles. A numerical simulation is carried out to analyze the proposed obstacle avoidance and path planning method. It is observed that the method finds a feasible path without collision for the multi-joint manipulator with a low computational cost. These results validated the effectiveness of the proposed method for path planning to avoid the obstacles.

INDEX TERMS FABRIK, multi-joint manipulator, obstacle avoidance, path planning, rapidly-exploring random tree.

I. INTRODUCTION

In the large-scale spacecraft applications (for example, on-orbit capture, auxiliary docking and on-orbit maintenance, etc.), more attention is paid on the space robot with a multi-joint manipulator [1], [2]. For these applications, it is crucial to find a method to capture a certain target using the manipulator. Therefore, a top priority research work is put on the manipulator's operational methods. In contrast to those of conventional manipulators with fewer joints, the multi-joint manipulator greatly increases the system's complexity, which makes the operation more difficult for the space robot. In addition, there may be obstacles in the workspace of the manipulator, in which case, a small collision can cause disaster. Therefore, the planning of a collision-free path from the initial position to the target position is the core theme.

Many methods and strategies of path planning for a manipulator have been proposed in literature. These literatures mainly focus on the two aspects: fixed-base and free-floating base. For the fixed-base, a path planning

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng H. Zhu.

technique is described by Falomir, E. for the robotic manipulators and mobile robots in the presence of stationary obstacles [3]. The method is about applying potential fields around configuration-space obstacles and using these fields to select a safe path for the robot to follow. For the manipulator path planning with obstacles, a novel and complete algorithm is designed by Bezzo, N. et al, which uses existing constrained optimization methods to generate optimal trajectories by taking it as a disjunctive program [4]. Ranjbar, B. et al propose an original solution and analytical comparison of the manipulator arms path planning by using artificial potential fields and multi-resolution occupancy grids [5]. For the 2R robot in a certain workspace with obstacles, Alekh, V. et al analyze the study and implementation of a path planning technique based on the potential field method [6]. For the motion planning of industrial manipulators, Akbaripour, H. et al and Lochan, K. et al design a new variation of sampling-based methods naming as the semi-lazy probabilistic roadmap relying on the advantages of the basic Probabilistic Road Map and lazy-PRM methods [7], [8]. For the free-floating base, Nakamura, Y. et al establishes a method based on the Lyapunov function for the non-holonomic motion planning of the space robot, where defines space vehicle with a 6-DOF (degrees of freedom) manipulator as a nine-variable system with six inputs [9]. A new path planning method for helping space manipulators in on-orbit mission avoiding obstacles based on an A* algorithm is designed by Hanxu and Shuangqi [10]. Xie, Y. et al propose a path planning algorithm based on Gauss–Newton iteration method for the motion path of space manipulators [11]. In summary, various approaches have been proposed to deal with the path planning problem of the manipulator, such as Enhanced Disturbance Map (EDM) [12], Least-Squares-Based Reaction Control (LSBRC) [13], Cyclic Arm Motion (CAM) [14], Reaction Null-Space (RNS) [15], etc. The above mentioned studies focus mainly on the path planning for the end-effector. However, all joints need to be considered during the movement of the end-effector. In this case, the inverse kinematics problem is also a key theme for path planning.

The inverse kinematics problem has been intensively studied in numerous literatures. For the computing numerical solutions to the inverse kinematics problem of robotic manipulators, a new method based on a combination of two nonlinear programming techniques and the forward recursion formulas, with the joint limitations of the robot being handled implicitly as simple boundary constraints is developed by Wang and Chen[16]. For inverse kinematics, Buss, S. R. introduces the Jacobian transpose method, the pseudo-inverse method, and the damped least squares method, and presents the mathematical foundations of these methods with an analysis based on the singular value decomposition [17]. For the general 6R robot, Qiao, S. et al introduces double quaternions and Dixon resultant to solve inverse kinematics analysis [18]. To solve the inverse kinematics, Rokbani, N. et al investigates the relative performances of PSO variants [19]. The Inverse Kinematics PSO (IK-PSO) is applied to a double link articulated system in this method. To solve the spatial n-R robot inverse kinematics problem, Wei, Y. et al designs a semi-analytic method and a general method [20]. For the inverse kinematics problem of robotic manipulators, Duka, A. V. investigates a method for finding a neuro-fuzzy [21]. To sum it up, various approaches have been proposed to deal with the inverse kinematics problem of the manipulator, such as pseudo-inverse of the Jacobian [22], Genetic Algorithms [23], [24], Neural Networks [25], Fuzzy Logic [26], etc. However, general methods of the inverse kinematics problem have high computational costs. Hence, the path planning for the manipulator in the space robot, that is, how to explore the entire workspace and find a feasible path with low computational costs is the impetus for this paper.

The Rapidly-Exploring Random Tree (RRT) algorithm has been intensively studied in numerous literatures. Vandeweghe, J. M. et, al propose a sampling-based path planning algorithm base on the RRT algorithm, its bootstraps is an optimal local controller based on the transpose of the Jacobian to a global RRT search [27]. Sintov, A. promotes the Time-Based RRT base on the RRT algorithms giving the nodes of the tree time parameters to make each node representing a specific state in a specific time denotes [28]. In addition, the application of the RRT algorithm for the collision-free path planning has been intensively studied. Rybus T. et, al. presents application of RRT algorithm for trajectory planning of the satellite-manipulator system [29]. João R.S. Benevides et, al. develops a collision-avoiding path planning for a free-floating planar manipulator [30]. The path planning algorithm use the RRT serves as the framework. And, a combination of the algorithm that reduces the metric sensitivity with a bidirectional approach is proposed in order to achieve a solution convergence. However, these algorithms have little discretion for the computational cost of the path planning algorithm.

This paper presents an algorithm for obstacle avoidance and path planning to ensure the safety of movement of the multi-joint manipulator in a workspace with obstacles. The algorithm is designed based on the RRT algorithm and the Forward and Backward Reaching Inverse Kinematics (FABRIK) algorithm [35]. Gratifyingly, the method can quickly find a feasible collision-free path with low computational cost for the multi-joint manipulator. The contributions can be highlighted as follows: 1) the RRT algorithm is introduced, under which the path planning algorithm can effectively explore the entire workspace, and greatly improve the possibility of searching for the obstacle avoidance paths; 2) in order to calculate the positions of the joints, based on the position of the end-effector, a method for the inverse kinematics problem is designed based on the FABRIK algorithm, under which the positions of each joint can be calculated with low computational cost.

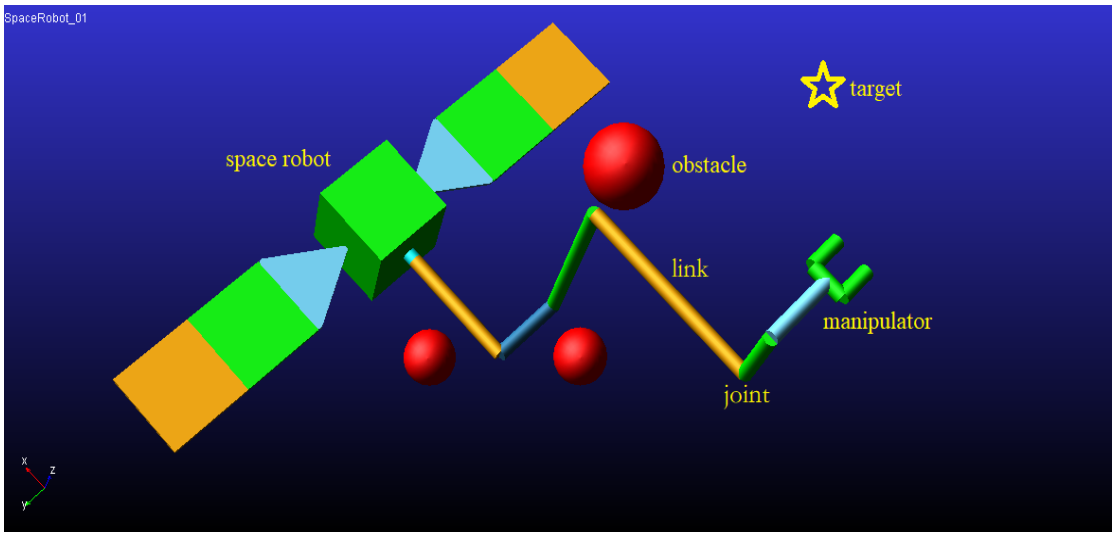


FIGURE 1. Schematic diagram of the operation of the space robot system.

This paper is organized as follows: Section 2 explains the kinematics equations of the multi-joint manipulator and the method of collision detection for the manipulator. An improved obstacle avoidance and path planning based on the RRT algorithm and the FABRIK algorithm for the multi-joint manipulator is designed in Section 3. Section 4 includes the simulation results and analyses. The conclusions are discussed in Section 5.

II. PROBLEM DESCRIPTION AND KINEMATICS MODELING

A. PROBLEM DESCRIPTION

The space robot system with multi-joint manipulator is equipped into a spacecraft with a n degree of freedom manipulator. The n degree of freedom manipulator is mounted on the spacecraft, the connection point between the spacecraft and the manipulator is the root joint of the manipulator. And, all the joints of the manipulator are revolute joints. As shown in FIGURE 1, the space robot system approaches and captures a target object in a space environment with obstacles. The mission of approaching and capturing the target object is accomplished by the end-effector of the manipulator, and the movement of the end-effector depends on the movement of the joints of the manipulator. In order to ensure the safety of the mission, the movement of the joints should guide the manipulator to avoid collision with the obstacles. Therefore, a collision-free path from the initial position to the target position is the basic guarantee to accomplish the mission. For this reason, this paper presents a path planning algorithm for the space robot with a multi-joint manipulator. In order to design the path planning algorithm, it is assumed that the positions of the obstacles and the position of the target object are known.

In this paper, the problem of obstacle avoidance in the operation of the space robot is described as a path planning problem: in the space environment with obstacles, a feasible path from the initial position to the target position is found for the end-effector of the manipulator, and there are no collisions with the obstacles in the process of the end-effector moving along the path. Since the end-effector movement depends on the joint movements of the manipulator, the path planning problem is changed into the following: Finding a feasible path in the joint space for each joint of the manipulator. As each joint of the manipulator moves along the path, the end-effector moves from the initial position to the target position, avoiding all possible collisions with the obstacles.

In order to complete the obstacle avoidance and path planning for the space robot with multi-joint manipulator, this paper presents an improved obstacle avoidance and path planning algorithm applied to the multi-joint manipulator. The improved algorithm combines the RRT algorithm and the FABRIK algorithm. The RRT algorithm can find a feasible path for the end-effector in the space environment with obstacles. However, the generation of each node in the random tree is required to solve the inverse kinematics problem of the multi-joint manipulator, which leads to a large amount of computation. The inverse kinematics problem of the multi-joint manipulator is to give out the positions of all joints based on the position of the end-effector and solve the joint variables. In this case, the FABRIK algorithm is introduced to solve the inverse kinematics problem. The algorithm can quickly determine all joint states of the multi-joint manipulator based on the position of the end-effector. Therefore, the improved path planning algorithm based on both the RRT algorithm and the FABRIK algorithm can quickly plan a feasible path for the multi-joint manipulator in the space environment with obstacles.

B. KINEMATICS EQUATION AND DYNAMICS EQUATION OF THE MANIPULATOR

The multi-joint manipulator of the space robot consists of links and joints, where the links are connected by the joints.

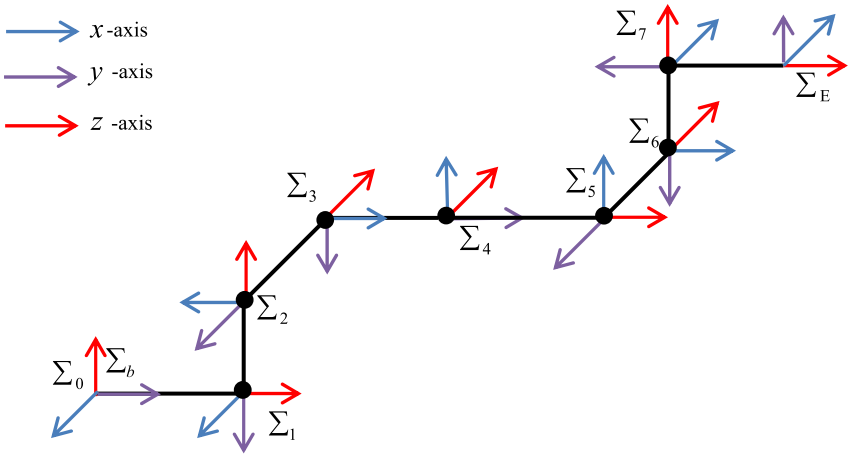


FIGURE 2. Inertial frame and joint frame for the space robot system.

In this paper, the multi-joint manipulator consists of M links and N joints, and the joints are connected in series. In this paper, to facilitate designing the algorithm, the series connecting rod manipulator is selected as the research object. Furthermore, suppose the number of the link is equal to the number of the joint ($M = N$). In fact, the number of the link may be unequal to the number of the joint of the manipulator. Because the FABRIK algorithm also applicable to solve the inverse kinematics problem of the parallel manipulator. Therefore, the algorithm designed in this paper is still usable, even if the number of the link is unequal to the number of the joint of the manipulator. In the space robot system, from the spacecraft to the end-effector, joints are defined as joint 1, joint 2, up to joint N, where the joint 1 connects the link 1 to the spacecraft. In this study, it is assumed that the links are straight links and the joints are revolute joints in the multi-joint manipulator.

The reference frames must be defined firstly before the kinematics equation is established. Based on the Denavit-Hartenberg (D-H) method, aiming at the multi-joint manipulator, the joint frames are created [31]. FIGURE 2 shows the joint frames. Unlike the general D-H reference frames, however, the joint frames in this paper are defined as follows: about the frame $\sum_i$ for the joint i, the origin of the frame $\sum_i$ coincides with the center point of the joint i, o_iz_i axis is along the joint axis of the joint i, o_ix_i axis is parallel to the common normal of o_iz_i axis of the joint i and $o_{i+1}z_{i+1}$ axis of the joint $i+1$, that is $o_ix_i = o_iz_i \times o_{i+1}z_{i+1}$, o_iy_i axis follows the right-hand rule. In addition, the inertial frame $\sum_0$ is defined at the center of mass of the spacecraft; the end-effector frame $\sum_E$ is defined at the end-effector of the manipulator.

It is worth noting that, to simplify algorithm design, the inertial frame $\sum_0$ is defined at the center of mass of the spacecraft in this paper. However, the origin of the inertial frame may not be at the center of the spacecraft. In this paper, the problem can be solved by introduce the rotation matrix. Using the rotation matrix, the variables can be represented in different coordinate systems. In this case, a body frame $\sum_b$ of the spacecraft is defined at the center of mass of the spacecraft. And, the rotation matrix $\boldsymbol{R}_{bi}$ from body frame $\sum_b$ to the inertial frame $\sum_0$ can be calculated by the position and the attitude of the spacecraft.

In fact, the position of the space robot and obstacles in space are changed real-time during the on-orbit operation. In addition, the earth centered inertial frame $\sum_I$ is defined at the center of earth. Then, suppose the position vector of the space robot in the earth centered inertial frame is r_b^I, and the position vector of obstacles are r_o^I. Conveniently, by the definition of the inertial frame $\sum_0$, the position vector of the origin of the inertial frame $\sum_0$ in the earth centered inertial frame $\sum_I$ is equal to the position vector r_b^I. Therefore, the relative position vector of the space robot to the origin of the inertial frame $\sum_0$ in the inertial frame $\sum_0$ is a constant value. As for the position change of obstacles, suppose the position change between the space robot and obstacles is minor. In this paper, the spherical-envelope method is used to deal with the minor change.

According to the D-H method, the joint frames of the adjacent joints of the manipulator can be overlapped by two translations and two rotations. That is, the joint frame $\sum_{i-1}$ coincides with joint frame $\sum_i$ by the following operation: rotate α_{i-1} in $o_{i-1}x_{i-1}$ axis $\rightarrow$ translate α_{i-1} by $o_{i-1}x_{i-1}$ axis $\rightarrow$ rotate θ_i in $o_{i-1}z_{i-1}$ axis $\rightarrow$ translate d_i by $o_{i-1}z_{i-1}$. In this paper, the homogeneous transformation matrix is used to represent the transformation between the joint frames. Thus, the homogeneous transformation matrix ${}^i\boldsymbol{T}_{i-1}$ from joint frame $\sum_{i-1}$ to joint frame $\sum_i$ can be expressed as follows:

$$
{}^i\boldsymbol{T}_{i-1} = \begin{bmatrix} \cos\theta_i & \sin\theta_i\cos\alpha_{i-1} & \sin\theta_i\sin\alpha_{i-1} & -a_{i-1}\cos\theta_i \\ -\sin\theta_i & \cos\theta_i\cos\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & a_{i-1}\sin\theta_i \\ 0 & -\sin\alpha_{i-1} & \cos\alpha_{i-1} & -d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}
$$

Similarly, the homogeneous transformation matrix ${}^{i-1}\boldsymbol{T}_i$ from joint frame $\sum_i$ to joint frame $\sum_{i-1}$ can be written as follows:

$$
{}^{i-1}\boldsymbol{T}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i\sin\alpha_{i-1} \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i\cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}
$$

Furthermore, the position $\boldsymbol{p}_i^0$ of the joint i in the inertial frame can be obtained by the continuous transformation of the frames:

$$
\boldsymbol{p}_i^0 = {}^0\boldsymbol{T}_i\boldsymbol{p}_i^i = {}^0\boldsymbol{T}_1{}^1\boldsymbol{T}_2\cdots{}^{i-1}\boldsymbol{T}_i\boldsymbol{p}_i^i \tag{3}
$$

where, the position $\boldsymbol{p}_i^i$ is the position of the joint i in the joint frame $\sum_i$, and $\boldsymbol{p}_i^i = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}}$.

Similarly, the position $\boldsymbol{p}_x^0$ in the inertial frame can be expressed in the joint frame $\sum_i$ of the joint i as follows:

$$
\boldsymbol{p}_x^i = {}^i\boldsymbol{T}_0\boldsymbol{p}_x^0 = {}^i\boldsymbol{T}_{i-1}\cdots{}^2\boldsymbol{T}_1{}^1\boldsymbol{T}_0\boldsymbol{p}_x^0 \tag{4}
$$

As shown in FIGURE 1, the space robot system is composed of a spacecraft and a multi-joint manipulator with n degree of freedom, therefore there is $n+1$ bodies in total. Then, the dynamic equations of the space robot system using the Lagrangian mechanism can be expressed as follows [32]:

$$
\begin{bmatrix} \boldsymbol{H}_b & \boldsymbol{H}_{bm} \\ \boldsymbol{H}_{bm}^T & \boldsymbol{H}_m \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{x}}_b \\ \ddot{\boldsymbol{\theta}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{c}_b \\ \boldsymbol{c}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_b \\ \boldsymbol{\tau}_m \end{bmatrix} \tag{5}
$$

where, $\boldsymbol{H}_b \in \mathbb{R}_{3\times3}$ is the inertia matrix of the spacecraft, $\boldsymbol{H}_{bm} \in \mathbb{R}_{3\times n}$ is the coupling inertia matrix, $\boldsymbol{H}_m \in \mathbb{R}_{n\times n}$ is the inertia matrix of the manipulator, $\ddot{\boldsymbol{x}}_b \in \mathbb{R}_{3\times1}$ is the vector of linear and angular accelerations of the spacecraft, $\ddot{\boldsymbol{\theta}} \in \mathbb{R}_{n\times1}$ is the joint accelerations of the manipulator, $\boldsymbol{c}_b \in \mathbb{R}_{3\times1}$, $\boldsymbol{c}_m \in \mathbb{R}_{n\times1}$ are the velocity dependent non-linear terms, $\boldsymbol{f}_b \in \mathbb{R}_{3\times1}$ is the force exerted on the spacecraft, $\boldsymbol{\tau}_m \in \mathbb{R}_{n\times1}$ is the torque exerted on the manipulator joints. Matrix expressions for $\boldsymbol{H}_b$, $\boldsymbol{H}_{bm}$, $\boldsymbol{H}_m$ can be found in literature [38]. For the matrix $\boldsymbol{H}_{bm}$, the angular momentum equation with zero initial angular momentum $\boldsymbol{L} = 0$ and zero attitude distance $\dot{\boldsymbol{\theta}} = 0$ becomes $\boldsymbol{H}_{bm}\dot{\boldsymbol{\theta}} = 0$. Moreover, the equation yields the following null-space solution $\dot{\boldsymbol{\theta}} = \left(\boldsymbol{E} - \boldsymbol{H}_{bm}^{+}\boldsymbol{H}_{bm}\right)\zeta$, where $(\cdot)^{+} = (\cdot)^{\mathrm{T}}\left((\cdot)(\cdot)^{\mathrm{T}}\right)^{-1}$ denotes the right pseudoinverse of $(\cdot)$, and noting that $(\cdot)(\cdot)^{+} = \boldsymbol{E}$, with $\boldsymbol{E}$ being an identity matrix of proper dimension. The joint motion given by the equation can ensure zero disturbance to the base attitude. The vector ζ is arbitrary and the null-space of the inertia matrix $\boldsymbol{H}_{bm}$ is called the reaction null-space. The expression $\boldsymbol{P} = \boldsymbol{E} - \boldsymbol{H}_{bm}^{+}\boldsymbol{H}_{bm}$ appearing in the equation denotes the projector onto the null-space of the coupled inertial matrix $\boldsymbol{H}_{bm}$.

For the free-floating space robot system, there is no external force and torque applied on the end-effectors and to the spacecraft. Therefore, the linear momentum $\boldsymbol{P}_0$ and angular momentum $\boldsymbol{L}_0$ of the whole robot system are conserved and can be expressed as follows:

$$
\begin{bmatrix} \boldsymbol{P}_0 \\ \boldsymbol{L}_0 \end{bmatrix} = \boldsymbol{H}_b\dot{\boldsymbol{x}}_b + \boldsymbol{H}_{bm}\dot{\boldsymbol{\theta}} \tag{6}
$$

Suppose the initial linear and angular momentum $\boldsymbol{P}_0 = \boldsymbol{L}_0 = 0$, since $\boldsymbol{H}_b$ is always invertible, the motion of the base can be described as follows:

$$
\dot{\boldsymbol{x}}_b = \begin{bmatrix} \dot{\boldsymbol{r}}_b \\ \boldsymbol{\omega}_b \end{bmatrix} = -\boldsymbol{H}_b^{-1}\boldsymbol{H}_{bm}\dot{\boldsymbol{\theta}} \tag{7}
$$

where, $\boldsymbol{r}_b$ is the position of the base, $\boldsymbol{\omega}_b$ is the attitude angular velocity of the base.

C. COLLISION DETECTION OF THE MANIPULATOR

In order to ensure the safety of the mission, the space robot system should avoid collision with the obstacles during operation. Therefore, the collision detection of the manipulator should be considered in the path planning. That is to say, it is necessary to consider the possibility of the joints and the links of the manipulator colliding with the obstacles in the space environment during its movement. In this paper, a collision detection method based on the enveloping principle of the space geometry is designed. In the collision detection method, the cylinder is used as the envelope of each link of the manipulator, and the sphere is used as the envelope of the obstacle. In relation to designing the enveloping method, the collision detection problem between the manipulator and the obstacles can be transformed into that the problem of the positional relationship between the cylinders and the spheres. Although the enveloping principle can simplify the collision detection problem, it also reduces the workspace of the manipulator to a given extent. Therefore, as much as is possible, the minimum diameter cylinders are used as the envelopes of the links of the manipulator, and the minimum diameter spheres are used as the envelopes of the obstacles.

The minimum diameter cylinder is used as the envelope of the link of the manipulator. According to the kinematics equations, this is the joint i on one end of the link i, and the joint $i+1$ on the other end, where, the positions of the joints are $\boldsymbol{p}_i^0 = \left(x_{pi}^0 w_{pi}^0, y_{pi}^0 w_{pi}^0, z_{pi}^0 w_{pi}^0, w_{pi}^0\right)$ and $\boldsymbol{p}_{i+1}^0 = \left(x_{pi+1}^0 w_{pi+1}^0, y_{pi+1}^0 w_{pi+1}^0, z_{pi+1}^0 w_{pi+1}^0, w_{pi+1}^0\right)$, respectively. Suppose the cylinder diameter of the envelope of the link i is $2r_{Li}$, then, the parametric equation of the spatial straight line of the center line of the cylinder can be written as follows:

$$
\frac{x - x_{pi}^0}{x_{pi+1}^0 - x_{pi}^0} = \frac{y - y_{pi}^0}{y_{pi+1}^0 - y_{pi}^0} = \frac{z - z_{pi}^0}{z_{pi+1}^0 - z_{pi}^0} \tag{8}
$$

Similarly, the minimum diameter sphere is used as the envelope of the obstacle. Suppose the central position of the obstacle j is $\boldsymbol{p}_{oj}^0 = \left(x_{oj}^0 w_{oj}^0, y_{oj}^0 w_{oj}^0, z_{oj}^0 w_{oj}^0, w_{oj}^0\right)$, and the sphere diameter of the envelope of the obstacle j is $2r_{oj}$, then, the parametric equation of the sphere can be written

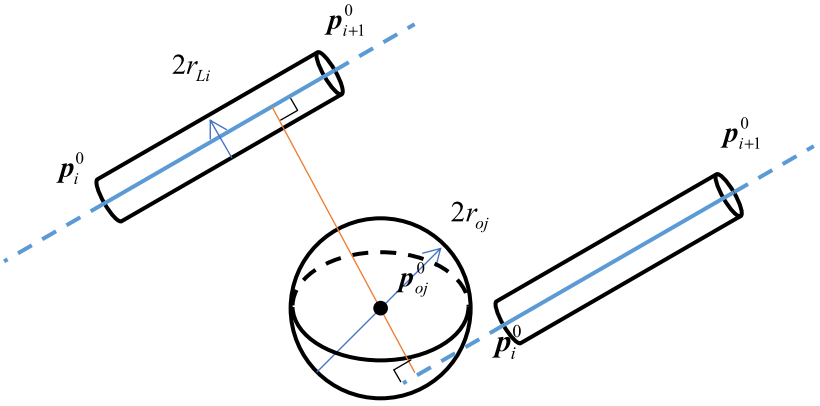


FIGURE 3. The positional relationship between the cylinder and the sphere.

as follows:

$$\left(x - x^0_{oj}\right)^2 + \left(y - y^0_{oj}\right)^2 + \left(z - z^0_{oj}\right)^2 = r^2_{oj} \tag{9}$$

The next step is to determine the positional relationship between the cylinder and the sphere, that is, compare the distance between the center line and the central position with the sum of the radius of the cylinder and the radius of the sphere. FIGURE 3 shows the positional relationship between the cylinder and the sphere. It can be seen that the minimum distance between the central position of the sphere and the center line of the cylinder is the length of the perpendicular. However, the intersection point of the perpendicular and the center line may fall on the center line or on the extended cord of the center line. Therefore, it is necessary to analyze the two cases separately. If the intersecting point is on the center line, the minimum distance d_{Lioj} between the central position of the sphere and the center line of the cylinder can be written as follows (10), as shown at the bottom of this page.

If the intersecting point is on the extended cord of the center line, the minimum distance d_{Lioj} between the central position of the sphere and the center line of the cylinder can be written as follows:

$$d_{Lioj} = \min\left\{\begin{array}{l}\sqrt{\left(x^0_{oj}-x^0_{pi}\right)^2+\left(y^0_{oj}-y^0_{pi}\right)^2+\left(z^0_{oj}-z^0_{pi}\right)^2}, \\ \sqrt{\left(x^0_{oj}-x^0_{pi+1}\right)^2+\left(y^0_{oj}-y^0_{pi+1}\right)^2+\left(z^0_{oj}-z^0_{pi+1}\right)^2}\end{array}\right\} \tag{11}$$

Definition F_{Lioj} is the result of the collision detection, and $F_{Lioj} = 1$ indicates that the link i collides with the obstacle j, $F_{Lioj} = 0$ indicates that the link i does not collide with the obstacle j. Thus,

$$F_{Lioj} = \begin{cases} 1 & d_{Lioj} \le r_{Li} + r_{oj} \\ 0 & d_{Lioj} > r_{Li} + r_{oj} \end{cases} \tag{12}$$

There are coupling between the manipulator and the spacecraft. The manipulator movement will lead the pose change of the spacecraft, so that the pose change affects the manipulator's planning precision. Therefore, it is important to deal with the pose change during the path planning. Here, the pose change of the spacecraft is equal with the position change of the obstacle. Since this paper is aiming at finding a collision-free path from the initial position to the target position for the multi-joint manipulator, the non-holonomic constraint problem transforms into a holonomic constraint problem. This equivalent operation has greatly simplified the design of the path planning algorithm.

Suppose the body frame of the spacecraft is $\sum_b$, and the initial position and orientation of the body frame $\sum_b(0)$ coincides with the inertial frame $\sum_0$.

According to Eq. (7), the pose of the spacecraft is $\boldsymbol{x}_b(t) = \left[x_b\ y_b\ z_b\ \theta\ \varphi\ \psi\right]^{\mathrm{T}}$ during the movement of the manipulator, where t is the time. Then, homogeneous transformation matrix ${}^b\boldsymbol{T}_b$ from the initial body frame $\sum_b(0)$ to the body frame $\sum_b(t)$ at time t can be expressed as follows:

$${}^b\boldsymbol{T}_b = \boldsymbol{R}\left(\theta \quad \varphi \quad \psi\right)\boldsymbol{T}\left(x_b \quad y_b \quad z_b\right) \tag{13}$$

where, $\boldsymbol{R}\left(\theta\ \varphi\ \psi\right)$ is the homogeneous transformation matrix of the Euler angles θ, φ, ψ, $\boldsymbol{T}\left(x_b\ y_b\ z_b\right)$ is the homogeneous transformation matrix of the translation x_b, y_b, z_b.

$$d_{Lioj} = \frac{\sqrt{\left|\begin{matrix} y^0_{oj}-y^0_{pi} & z^0_{oj}-z^0_{pi} \\ y^0_{pi+1}-y^0_{pi} & z^0_{pi+1}-z^0_{pi}\end{matrix}\right| + \left|\begin{matrix} z^0_{oj}-z^0_{pi} & x^0_{oj}-x^0_{pi} \\ z^0_{pi+1}-z^0_{pi} & x^0_{pi+1}-x^0_{pi}\end{matrix}\right| + \left|\begin{matrix} x^0_{oj}-x^0_{pi} & y^0_{oj}-y^0_{pi} \\ x^0_{pi+1}-x^0_{pi} & y^0_{pi+1}-y^0_{pi}\end{matrix}\right|}}{\sqrt{\left(x^0_{pi+1}-x^0_{pi}\right)^2+\left(y^0_{pi+1}-y^0_{pi}\right)^2+\left(z^0_{pi+1}-z^0_{pi}\right)^2}} \tag{10}$$

Then, the position of the obstacle j at time t can be expressed as follows:

$$\boldsymbol{p}_{oj}^{t} = {}^{b}\boldsymbol{T}_{b}\boldsymbol{p}_{oj}^{0} = \boldsymbol{R}\begin{pmatrix}\theta & \varphi & \psi\end{pmatrix}\boldsymbol{T}\begin{pmatrix}x_b & y_b & z_b\end{pmatrix}\boldsymbol{p}_{oj}^{0} \tag{14}$$

III. PATH PLANNING ALGORITHM OF OBSTACLE AVOIDANCE FOR THE MANIPULATOR

In this section, firstly, the problem of obstacle avoidance and path planning is described by a mathematical model. Secondly, the steps of the obstacle avoidance and path planning algorithm are established on the principle of the RRT algorithm, and the method for generating random points by artificial guidance is introduced for higher searching speeds. Finally, the FABRIK algorithm is introduced to solve the inverse kinematics problems of the multi-joint manipulator.

A. MATHEMATICAL MODEL FOR THE PROBLEM OF OBSTACLE AVOIDANCE AND PATH PLANNING

In this paper, the problem of obstacle avoidance and path planning is described by a mathematical model. According to the Section 2.1, the problem of obstacle avoidance and path planning is described for each joint of the manipulator, whose a feasible path is found in the joint space. As each joint of the manipulator moves along the path, the end-effector moves from the initial position to the target position, and there are no collisions with the obstacles.

In this paper, it is assumed that:

1) There are m obstacles in the inertial space S, and the center positions of the obstacles are $\boldsymbol{p}_{oj}^{0}, j = 1, 2, \cdots, m$, the sphere radiuses of the envelopes of the obstacles are $r_{oj}, j = 1, 2, \cdots, m$;

2) The multi-joint manipulator consists of M links and N joints ($M = N$), and the ranges of movement of the joints are $[\theta_{i\min}, \theta_{i\max}], i = 1, 2, \cdots n$;

3) The initial joint angles of the multi-joint manipulator are $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_s = (\theta_1(0), \theta_2(0), \cdots \theta_N(0))$, the initial joint positions are $\boldsymbol{p}_i^0(0) = \boldsymbol{p}_{is}^0, i = 1, 2, \cdots, N$, the initial position of the end-effector is $\boldsymbol{p}_E^0(0) = \boldsymbol{p}_{Es}^0$, and the position of the target object is $\boldsymbol{p}_{Ed}^0$.

In relation to the kinematics equations of the multi-joint manipulator, the position of the end-effector depends on the joint angles. That is, the object of the planning is the position of the end-effector equal to the target position, thus,

$${}^{0}\boldsymbol{T}_{E}(\boldsymbol{\theta}(t_s))\boldsymbol{p}_{E}^{E} = \boldsymbol{p}_{Ed}^{0} \tag{15}$$

where, t_s is the time of the end-effector moves from the initial position to the target position, ${}^{0}\boldsymbol{T}_{E}(\boldsymbol{\theta}(t_s))$ is the homogeneous transformation matrix of the frame $\sum_E$ to frame $\sum_0$ in the final time. Considering the obstacle avoidance, the manipulator cannot collide with the obstacles during the movement of the joints. That is, the constraints of the planning is collision-free during the movement of the manipulator, thus,

$$F_{\text{sum}}(t) = \sum_{i=1}^{N}\sum_{j=1}^{m} F_{Lioj}(t) = 0, \quad 0 \le t \le t_s \tag{16}$$

where, $F_{Lioj}(t)$ is the result of the collision detection at time t. Considering the movement performances of the joints, then,

$$\boldsymbol{\theta}_{\min} \le \boldsymbol{\theta}(t) \le \boldsymbol{\theta}_{\max} \tag{17}$$

where, $\boldsymbol{\theta}_{\min} = (\theta_{1\min}, \theta_{2\min}, \cdots, \theta_{N\min})$, $\boldsymbol{\theta}_{\max} = (\theta_{1\max}, \theta_{2\max}, \cdots, \theta_{N\max})$. Therefore, the problem of obstacle avoidance and path planning algorithm generates the joint movements, which satisfies the equality and the inequality constraints, and can be written as:

$$\begin{cases}\boldsymbol{\theta}(0) = \boldsymbol{\theta}_s & {}^{0}\boldsymbol{T}_{E}(\boldsymbol{\theta}(t_s))\boldsymbol{p}_{E}^{E} = \boldsymbol{p}_{Ed}^{0} \\ \boldsymbol{\theta}_{\min} \le \boldsymbol{\theta}(t) \le \boldsymbol{\theta}_{\max} & F_{\text{sum}}(t) = 0\end{cases} \tag{18}$$

B. OBSTACLE AVOIDANCE AND PATH PLANNING USING THE RRT ALGORITHM

For the problem of obstacle avoidance and path planning shown in Eq. (18), a path planning algorithm is designed based on the RRT algorithm. The RRT algorithm generates the random points in the entire workspace, and the random tree is expanded based on the random points. Therefore, the algorithm can effectively explore the entire workspace, and greatly improve the possibility of searching for the obstacle avoidance paths.

The principle of the RRT algorithm is as follows: First, the initial point is used as a root node, and then the root node is expanded into a random tree by randomly adding the leaf nodes. Once the leaves in the random tree contain the target point or enter the target area, a path from the initial point to the target point can be found in the random tree. Based on the principle of the RRT algorithm, the proposed algorithm can be divided into three parts: Start, Tree extension, and Planning [33], [34], FIGURE 4 shows the schematic diagram of the RRT algorithm.

Based on the principle of the RRT algorithm, we design the obstacle avoidance and path planning algorithm for the multi-joint manipulator. FIGURE 5 shows the Flow diagram of the obstacle avoidance and path planning algorithm. The algorithm steps are as follows:

Input (Initial Parameters): The center positions $\boldsymbol{p}_{oj}^{0}, j = 1, 2, \cdots, m$ of the obstacles, the sphere radiuses $r_{oj}, j = 1, 2, \cdots, m$ of the envelopes of the obstacles, initial joint angles $\boldsymbol{\theta}(0) = \boldsymbol{\theta}_s = (\theta_1(0), \theta_2(0), \cdots, \theta_N(0))$, the initial joint positions $\boldsymbol{p}_i^0(0) = \boldsymbol{p}_{is}^0, i = 1, 2, \cdots N$, the initial position $\boldsymbol{p}_{Es}^0$ of the end-effector, and the position $\boldsymbol{p}_{Ed}^0$ of the target object.

Step 1 (Initialization): Begins the random tree with the initial position $\boldsymbol{p}_{Es}^0$ as the root node, and the random tree is recorded as *Tree*;

Step 2: Generate the new node. A random point $\boldsymbol{p}_{rand}^0$ is randomly generated in the inertia space, and the nearest node $\boldsymbol{p}_{near}^0$ closest to the random point $\boldsymbol{p}_{rand}^0$ is found on the random

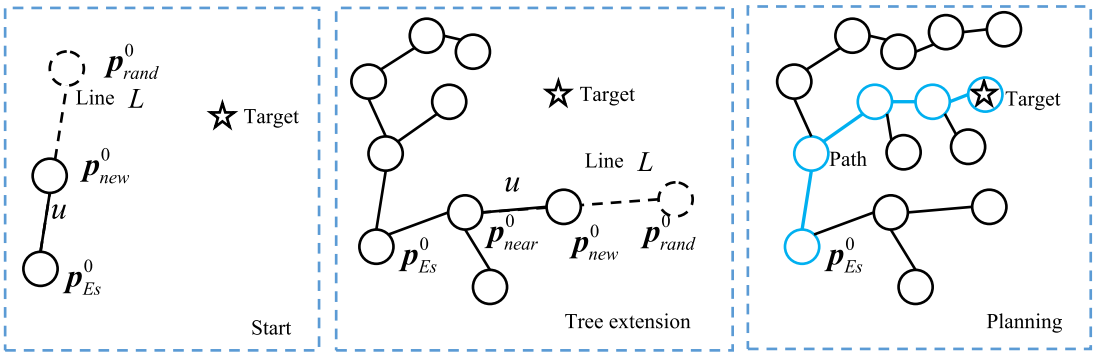


FIGURE 4. Schematic diagram of the RRT algorithm.

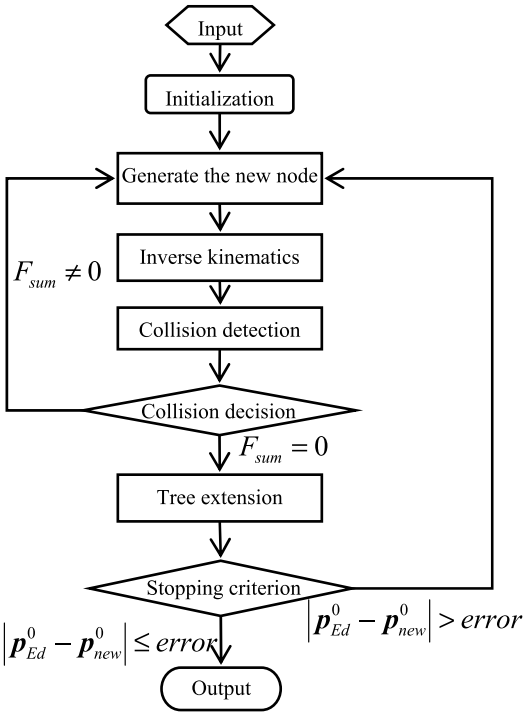


FIGURE 5. Flow diagram of the obstacle avoidance and path planning algorithm.

tree *Tree*, and then the new node p^0_{new} is determined on the connecting line between the near node p^0_{near} and the random point p^0_{rand}, and the new node p^0_{new} satisfies the equation $|p^0_{new} - p^0_{near}| = u$, where, u is the control variable, $|\bullet|$ is the distance between the point p^0_{near} and the point p^0_{new}, and the distance is calculated in Euclidean space;

Note: the random point is generated in the whole space.

Step 3: Inverse kinematics problem of the multi-joint manipulator. The new node p^0_{new} is used as the position of the end-effector of the manipulator, and then the joint angles θ_{new} and the joint positions p^0_{inew} of the manipulator are calculated as an inverse kinematics problem;

Step 4: Collision detection. Based on the collision detection method shown in Section 2.3, there is a determination of whether there is a collision with an obstacle during the movement of the end-effector from the position p^0_{near} to the position p^0_{new} (or the movement of the joint from the joint angle θ_{near} to joint angle θ_{new}, or the movement of the joint from the joint position p^0_{inear} to the joint position p^0_{inew}), and subsequently the detection result is recorded as F_{sum};

Step 5: Collision decision. If the new node p^0_{new} meets the condition of the obstacle avoidance, that is $F_{\text{sum}} = 0$, then it proceeds to Step 6; if the new node p^0_{new} fails to meet the condition of the obstacle avoidance, that is $F_{\text{sum}} \neq 0$, then go it returns Step 2;

Step 6: Tree extension. The nearest node p^0_{near} is used as the parent node, and the new node p^0_{new} is added as the child node to the random tree *Tree*. The random tree *Tree* is then updated;

Step 7: Stopping criterion for iteration. This Step determines whether the new node p^0_{new} is the target position p^0_{Ed}, or near the target position p^0_{Ed}, that is, if $|p^0_{Ed} - p^0_{new}| \leq error$, then consequently ends the iteration; if $|p^0_{Ed} - p^0_{new}| > error$, then it returns Step 2;

Output: Obstacle avoidance path. Starting from the target node p^0_{Ed}, the corresponding parent node in the random tree is searched in turn until the initial node p^0_{Es}. The obtained nodes then represent the path point of the planned obstacle avoidance path.

From these algorithm steps, the pseudo-code of the RRT algorithm is as follows:

According to the principle of the RRT algorithm, this method adopts a random search. In this case, the algorithm can effectively explore the entire workspace, however, the algorithm has low computational efficiency. Therefore, the method of generating random points by artificial guidance is introduced for higher searching speeds in this paper. The specific operation of the method is as follows: when the RRT algorithm generates the random point p^0_{rand}, the target point p^0_{goal} is selected with a certain probability as the random point p^0_{rand} in the tree extension, that is $p^0_{rand} = p^0_{goal}$. The random point p^0_{rand} is equivalent to giving a direction of expansion in the random tree expansion, and using the target point p^0_{goal} as the random point with a certain probability is equivalent to driving the random tree to expand toward the target object.

Algorithm RRT Algorithm

1 Initialize: the root node of the random tree $\boldsymbol{p}_{Es}^0$, number of the iterations K, the target position $\boldsymbol{p}_{Ed}^0$
2 for $k = 1$ to K do
3 randomly generate $\boldsymbol{p}_{rand}^0$ in the inertia space
4 search $\boldsymbol{p}_{near}^0$ in the random tree *Tree*
5 generate $\boldsymbol{p}_{new}^0$ using the equation $|\boldsymbol{p}_{new}^0 - \boldsymbol{p}_{near}^0| = u$
6 calculate the joint angels $\boldsymbol{\theta}_{new}$ and the joint positions $\boldsymbol{p}_{inew}^0$
7 calculate the detection result F_{sum}
8 if $F_{sum} \neq 0$
9 return to 3
10 end
11 generate the new random tree *Tree*
12 calculate the stopping criterion for iteration $|\boldsymbol{p}_{Ed}^0 - \boldsymbol{p}_{new}^0|$
13 if $|\boldsymbol{p}_{Ed}^0 - \boldsymbol{p}_{new}^0| > error$
14 return to 3
15 else
16 break;
17 end
18 return the random tree *Tree*

In this, Step 2 in the above algorithm can be improved as follows:

Improved Step 2 (Generate the New Node): Generate a random number p, if $p \leq p_{goal}$, then $\boldsymbol{p}_{rand}^0 = \boldsymbol{p}_{Ed}^0$; if $p > p_{goal}$, then a random point $\boldsymbol{p}_{rand}^0$ is randomly generated in the inertia space, then the node $\boldsymbol{p}_{new}^0$ closest to the random point $\boldsymbol{p}_{rand}^0$ is found on the random tree *Tree*, and then the new node $\boldsymbol{p}_{new}^0$ is determined on the connecting line between the near node $\boldsymbol{p}_{near}^0$ and the random point $\boldsymbol{p}_{rand}^0$, and the new node $\boldsymbol{p}_{new}^0$ satisfies the equation $|\boldsymbol{p}_{new}^0 - \boldsymbol{p}_{near}^0| = u$, where, u is the control variable;

For the Improved Step 2, the pseudo-code of the step 3 of the RRT algorithm can be rewritten as follows:

Algorithm Improved Step 2 Algorithm

1 Input: the probability of selecting the target point p_{goal}, the target point $\boldsymbol{p}_{Ed}^0$ and the random tree *Tree*
2 generate a random number p
3 if $p \leq p_{goal}$
4 $\boldsymbol{p}_{rand}^0 = \boldsymbol{p}_{Ed}^0$
5 else
6 Generate a random point $\boldsymbol{p}_{rand}^0$
7 end
8 search $\boldsymbol{p}_{near}^0$ in the random tree *Tree*
9 generate $\boldsymbol{p}_{new}^0$ using the equation $|\boldsymbol{p}_{new}^0 - \boldsymbol{p}_{near}^0| = u$
10 return the new node $\boldsymbol{p}_{new}^0$

C. FABRIK ALGORITHM FOR THE INVERSE KINEMATICS PROBLEM

In Section 3.2, the path planning algorithm is designed based on the RRT algorithm for the multi-joint manipulator. The new nodes in the RRT algorithm are the randomly generated positions of the end-effector; therefore, the joint positions and joint angles need to be calculated according to the positions of the end-effector. That is to say, the inverse kinematics problem of the multi-joint manipulator exists in each node operation. As a result, the inverse kinematics problems will lead to several challenges such as poor computation efficiency due to large computation amount, and singularity. To curb these problems, the FABRIK algorithm is introduced in this paper. The FABRIK algorithm is first proposed by Aristidou, A. et al in computer graphics [35]. In literature [35], the FABRIK algorithm addresses the problem of manipulating articulated figures in an interactive and intuitive fashion for the design and control of their posture. In this paper, the FABRIK algorithm is improved to make it suitable for the multi-joint manipulator, and all the joints of the manipulator are revolute joints. The improved FABRIK algorithm is a novel heuristic method; it avoids the use of rotational angles or matrices, and instead finds each joint position by locating a point on a line, and thus, has low computational costs.

The FABRIK algorithm is divided into two stages: The first stage starts from the end-effector and moves inwards to the root joint, and finds each joint position successively along the link; the second stage stars from the root joint and moves outwards to the end-effector, and finds each joint position successively along the link. FIGURE 6 shows the Algorithm diagram of the FABRIK algorithm.

In general, the operations of above noted stages can be repeated until the position of the end-effector is the target position (or a position near the target). However, considering the movement constraints of the revolute joints and the random characteristics of the RRT algorithm, only one operation (one first stage and one second stage) is performed for the inverse kinematics problem of each node, and the new position of the end-effector after one operation is used as the new node in the random tree.

The FABRIK algorithm is used to solve the inverse kinematics problem in Step 3 of the obstacle avoidance and path planning algorithm. The planning algorithm shows that: the position of the joint i is $\boldsymbol{p}_i^0$, the joint angle of the joint i is θ_i, the new node $\boldsymbol{p}_{new}^0$ is the target position of the end-effector of the multi-joint manipulator. The algorithm is to find the position $\boldsymbol{p}_{inew}^0$ of the joint i and the joint angle of the joint i, if the end-effector at the position of the target, where, $i = 1, 2, \cdots, n$.

First, the new position $\boldsymbol{p}_i'^0$ of the joint i is estimated by the first stage. Here, we have:

$$\begin{cases} \boldsymbol{p}_E'^0 = \boldsymbol{p}_{new}^0 \\ \boldsymbol{p}_N'^0 = \boldsymbol{p}_E'^0 + \dfrac{\boldsymbol{p}_N^0 - \boldsymbol{p}_E'^0}{|\boldsymbol{p}_N^0 - \boldsymbol{p}_E'^0|} \cdot l_N \\ \boldsymbol{p}'^0_{N-1} = \boldsymbol{p}_N'^0 + \dfrac{\boldsymbol{p}_{N-1}^0 - \boldsymbol{p}_N'^0}{|\boldsymbol{p}_{N-1}^0 - \boldsymbol{p}_N'^0|} \cdot l_{N-1} \\ \quad \vdots \\ \boldsymbol{p}'^0_1 = \boldsymbol{p}_2'^0 + \dfrac{\boldsymbol{p}_1^0 - \boldsymbol{p}_2'^0}{|\boldsymbol{p}_1^0 - \boldsymbol{p}_2'^0|} \cdot l_1 \end{cases} \tag{19}$$

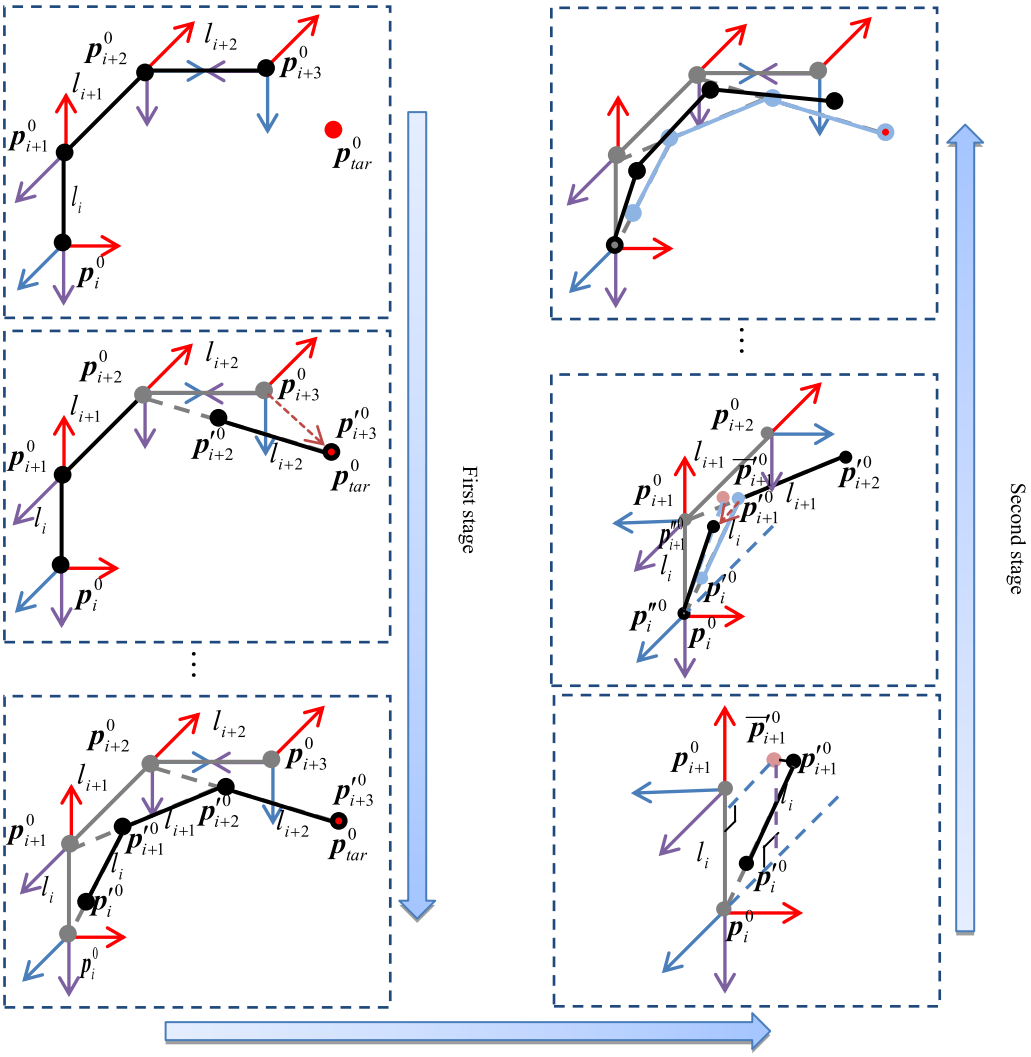


FIGURE 6. Algorithm diagram of the FABRIK algorithm.

where, $\boldsymbol{p}_E'^0$ is the new position of the end-effector, $\boldsymbol{p}'^0_1$ is the new position of the root joint.

Secondly, the new position $\boldsymbol{p}'^0_i$ of the joint i is estimated by the second stage. Let the new position $\boldsymbol{p}''^0_1$ of the root joint equal to the initial position $\boldsymbol{p}^0_1$ of the root joint, this yields:

$$\boldsymbol{p}''^0_1 = \boldsymbol{p}^0_1 \tag{20}$$

Then, in the joint frame $\sum_1$, the position $\boldsymbol{p}'^0_2$ of the joint 2 can be written as:

$$\boldsymbol{p}'^1_2 = {}^1\boldsymbol{T}_0 \boldsymbol{p}'^0_2 \tag{21}$$

And, the projection $\bar{\boldsymbol{p}}'^1_2$ of the position $\boldsymbol{p}'^0_2$ of the joint 2 in the $x_1o_1y_1$ plane of the joint frame $\sum_1$ can be written as:

$$\bar{\boldsymbol{p}}'^1_2 = \boldsymbol{A} \cdot \boldsymbol{p}'^1_2 \tag{22}$$

where,

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{23}$$

Then, in the inertial frame $\sum_0$, the projection $\bar{\boldsymbol{p}}'^0_2$ of the position $\boldsymbol{p}'^0_2$ of the joint 2 can be written as:

$$\bar{\boldsymbol{p}}'^0_2 = {}^0\boldsymbol{T}_1 \cdot \bar{\boldsymbol{p}}'^1_2 \tag{24}$$

Furthermore, the new position $\boldsymbol{p}''^0_2$ of the joint 2 can be written as:

$$\boldsymbol{p}''^0_2 = \boldsymbol{p}''^0_1 + \frac{\bar{\boldsymbol{p}}'^0_2 - \boldsymbol{p}''^0_1}{|\bar{\boldsymbol{p}}'^0_2 - \boldsymbol{p}''^0_1|} \cdot l_1 \tag{25}$$

And, the joint angle $\theta_{1\text{new}}$ of the joint 1 can also be expressed as:

$$\theta_{1new} = \theta_1 + \arctan\left(\frac{\bar{p}'^1_{2x}}{\bar{p}'^1_{2y}}\right) \tag{26}$$

where, $\bar{\boldsymbol{p}}'^1_2 = \left(\bar{p}'^1_{2x}, \bar{p}'^1_{2y}, \bar{p}'^1_{2z}\right)$.

In conclusion,

$$\begin{cases} \boldsymbol{p}''^0_1 = \boldsymbol{p}^0_1 \\ \boldsymbol{p}''^0_2 = \boldsymbol{p}''^0_1 + \dfrac{\boldsymbol{p}'^0_2 - \boldsymbol{p}''^0_1}{\left|\boldsymbol{p}'^0_2 - \boldsymbol{p}''^0_1\right|} \cdot l_1 \\ \quad\vdots \\ \boldsymbol{p}''^0_N = \boldsymbol{p}''^0_{N-1} + \dfrac{\boldsymbol{p}'^0_N - \boldsymbol{p}''^0_{N-1}}{\left|\boldsymbol{p}'^0_N - \boldsymbol{p}''^0_{N-1}\right|} \cdot l_{N-1} \\ \boldsymbol{p}''^0_E = \boldsymbol{p}''^0_N + \dfrac{\boldsymbol{p}'^0_E - \boldsymbol{p}''^0_N}{\left|\boldsymbol{p}'^0_E - \boldsymbol{p}''^0_N\right|} \cdot l_N \end{cases} \tag{27}$$

$$\theta_{inew} = \theta_i + \arctan\left(\frac{\bar{p}'^i_{i+1x}}{\bar{p}'^i_{i+1y}}\right) \tag{28}$$

where,

$$\bar{\boldsymbol{p}}'^0_i = {}^0\boldsymbol{T}_i \boldsymbol{A}^i \boldsymbol{T}_0 \boldsymbol{p}'^0_i, \quad i = 1, 2, \cdots, N, E \tag{29}$$

After one operation (one first stage and one second stage), the new position $\boldsymbol{p}''^0_E$ of the end-effector is used as the new node $\boldsymbol{p}^0_{new}$, that is:

$$\boldsymbol{p}^0_{new} = \boldsymbol{p}''^0_E, \quad \boldsymbol{p}^0_{inew} = \boldsymbol{p}''^0_i \tag{30}$$

Therefore, Step 3 in the obstacle avoidance and path planning algorithm can be improved as follows:

Improved Step 3: Inverse kinematics problem of the multi-joint manipulator. The new node $\boldsymbol{p}^0_{new}$ is used as the position of the end-effector of the manipulator, and then the joint angles $\boldsymbol{p}^0_{i+3}$ and the joint positions $\boldsymbol{p}^0_{inew}$ of the manipulator are calculated by the FABRIK algorithm (Eq. (19), Eq. (27), Eq. (28), Eq. (30));

IV. SIMULATION RESULTS

A. SIMULATION

Numerical simulation results of the FABRIK algorithm and the obstacle avoidance and path planning algorithm were proposed in this section. The numerical simulation is accomplished by Matlab software. The D-H parameters of the multi-joint manipulator are shown in TABLE 1, and FIGURE 7 shows the multi-joint manipulator. In FIGURE 7, the blue filled circles are the joints of the manipulator, the red asterisk is the end-effector of the manipulator, the blue lines are the links of the manipulator, and the green line is the distance from the center of mass of the spacecraft to the joint 1 of the manipulator. In addition, the dynamics parameters of the multi-joint manipulator are shown in TABLE 2.

The initial position $\boldsymbol{p}^0_{Es}$ of the end-effector and the position $\boldsymbol{p}^0_{Ed}$ of the target object are:

$$\boldsymbol{p}^0_{Es} = \begin{pmatrix} -0.86 & 1.81 & 0.83 & 1 \end{pmatrix} \text{ m}$$
$$\boldsymbol{p}^0_{Ed} = \begin{pmatrix} -0.26 & 0.28 & 0.72 & 1 \end{pmatrix} \text{ m}$$

TABLE 1. D-H parameters of the multi-joint manipulator.

i	a_{i-1} (m)	α_{i-1} (deg)	d_i (m)	θ_i (deg)
1	0	-90	0.7	0
2	0	90	0.43	-90
3	0	90	0.43	180
4	0.38	0	0	-90
5	0	-90	0.43	0
6	0	90	0.43	90
7	0	90	0.4	90
End-effector	0	90	0.3	0

TABLE 2. Dynamics parameters of the multi-joint manipulator.

i	Mass (kg)	Moment of inertia $(\text{kg}\cdot\text{m}^2)$		
		I_{xx}	I_{yy}	I_{zz}
Spacecraft	1000	1200	1200	1200
1	20.76	0.645	0.152	0.645
2	20.76	0.645	0.152	0.645
3	40.45	0.284	1.312	1.308
4	31.15	0.137	1.011	0.013
5	20.76	0.645	0.645	0.152
6	20.76	0.645	0.645	0.152
7	27.96	1.311	1.311	0.22

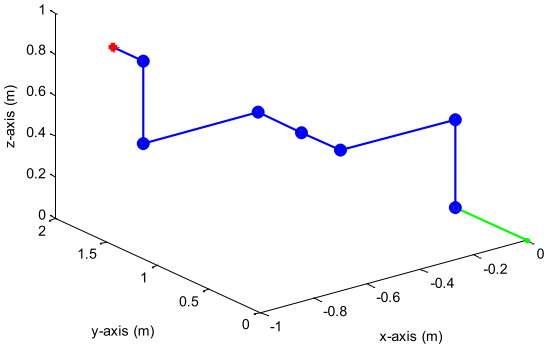


FIGURE 7. The multi-joint manipulator with the initial parameters.

In order to verify the assumption of the FABRIK algorithm, the position $\boldsymbol{p}^0_{Ed}$ of the target object is used as the final position of the end-effector. Then, the positions of the joints using the FABRIK algorithm are calculated. The results of the performance of different operations (one operation represents one first stage and one second stage) for the inverse kinematics problem are shown in FIGURE 8.

In FIGURE 8, the green five-pointed star is the position $\boldsymbol{p}^0_{Ed}$ of the target object. FIGURE 8 shows that the end-effector gets closer to the target object as the number of the operations increases. In FIGURE 8, it can be seen that the position of the end-effector is $\boldsymbol{p}^0_E = \begin{pmatrix} -0.4227 & 0.7125 & 1.0273 & 1 \end{pmatrix}$ m after 1 operation, $\boldsymbol{p}^0_E = \begin{pmatrix} -0.3143 & 0.3832 & 0.8025 & 1 \end{pmatrix}$ m after 10 operations, and $\boldsymbol{p}^0_E = \begin{pmatrix} -0.2773 & 0.3069 & 0.6910 & 1 \end{pmatrix}$ m after 100 operations. Comparing the results of the different operations it is observed that: relative to 1 operation, the final position of the end-effector of 10 operations is closer to the target position. However, compared with 10 operations,

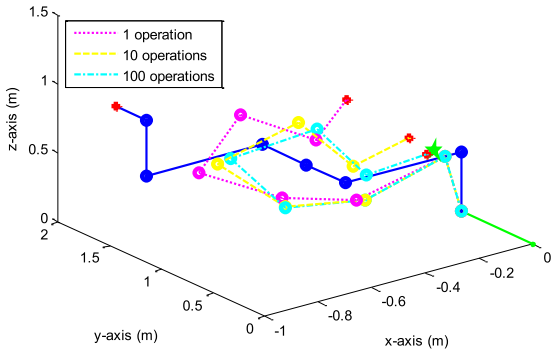


FIGURE 8. Results of different operations for the inverse kinematics problem.

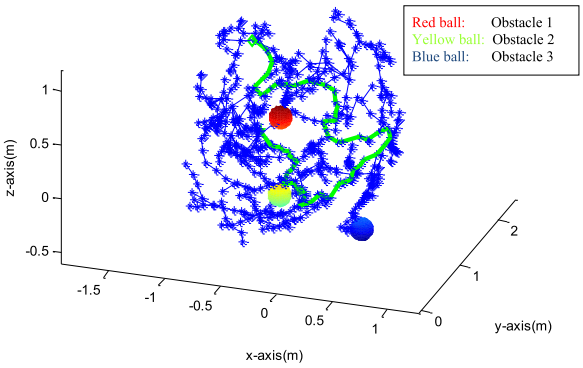


FIGURE 9. All leaf nodes of the Random Tree in the RRT algorithm.

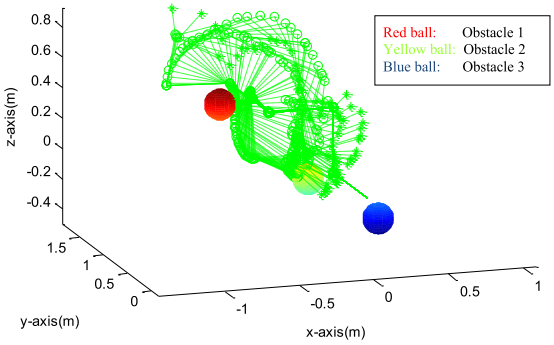


FIGURE 10. Planned path of the manipulator.

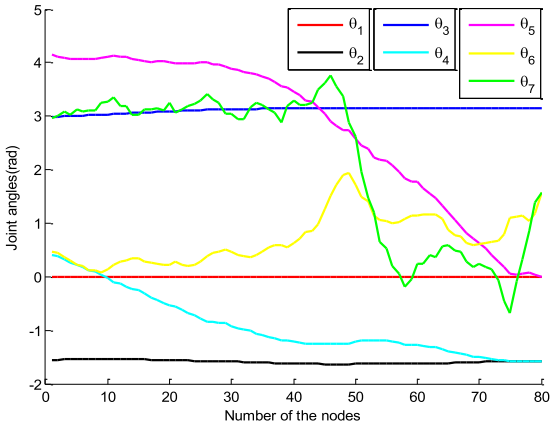


FIGURE 11. Planned paths of the joints angles of the manipulator.

the difference is marginal, although the final position of the end-effector of 100 operations is closer to the target position. That is to say, the inverse kinematics problem can be calculated by the FABRIK algorithm using fewer operations. Therefore, the joints positions can be found with low computational cost by the end-effector using the FABRIK algorithm. Considering that the workspace exploration of the RRT algorithm is random, only one operation (one first stage and one second stage) is performed for the inverse kinematics problem of each node in the obstacle avoidance and path planning algorithm.

In fact, in order to verify the assumption of the obstacle avoidance and path planning algorithm, two cases are assumed as follows:

Case 1: The method of generating a random point by artificial guidance is not introduced for the RRT algorithm (Step 2 is used in the obstacle avoidance and path planning algorithm).

Case 2: The method of generating a random point by artificial guidance is introduced for the RRT algorithm (Improved Step 2 is used in the obstacle avoidance and path planning algorithm).

In both cases, the control variable u, and the cylinder diameter of the envelope of the link $2r_{Li}$ are the same. Also the central position of the obstacle $\boldsymbol{p}_{oj}^{0}$, and the sphere diameter of the envelope of the obstacle $2r_{oj}$ are expressed as:

$$\begin{aligned} u &= 0.2\text{m}, \quad r_{Li} = 0.01\text{m}, \ i = 1, 2, \cdots, 7 \\ \boldsymbol{p}_{o1}^{0} &= \begin{pmatrix} -0.2 & 0.6 & 0 & 1 \end{pmatrix}\text{m}, \quad r_{o1} = 0.1\text{m} \\ \boldsymbol{p}_{o2}^{0} &= \begin{pmatrix} 0.4 & 1 & -0.4 & 1 \end{pmatrix}\text{m}, \quad r_{o2} = 0.1\text{m} \\ \boldsymbol{p}_{o3}^{0} &= \begin{pmatrix} -0.5 & 1.5 & 0.3 & 1 \end{pmatrix}\text{m}, \quad r_{o3} = 0.1\text{m} \end{aligned}$$

For the probability p_{goal}, the RRT algorithm has low computational efficiency if the parameter value of the probability is too small but if it is too big, the RRT algorithm cannot effectively explore the entire workspace. In Case 2, based on the numerical simulation results, the probability p_{goal} that use the target point X_{goal} as the random point X_{rand} in the tree extension is:

$$p_{goal} = 0.01$$

The above parameters are used for numerical simulations. In Case 1, using the obstacle avoidance and path planning algorithm, the results are shown in FIGURE 9-FIGURE 12. In Case 2, where the method of generating a random point by artificial guidance is introduced for the RRT algorithm, using the obstacle avoidance and path planning algorithm, the results are shown in FIGURE 13-FIGURE 16.

In the above-mentioned results, we get the collision-free path of the multi-joint manipulator using the obstacle avoidance and path planning algorithm. In FIGURE 9 and FIGURE 13, the spheres are the obstacles in the workspace;

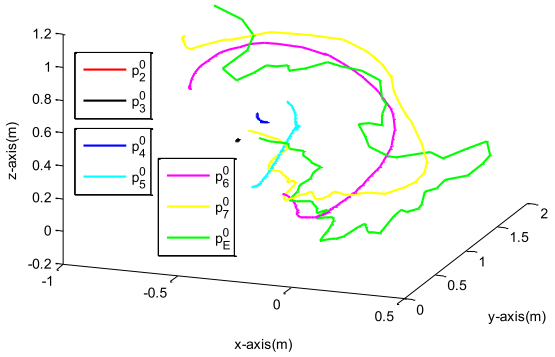


FIGURE 12. Planned paths of the positions of the joints of the manipulator.

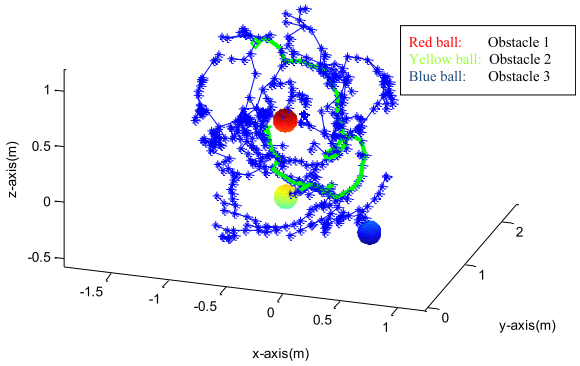


FIGURE 13. All leaf nodes of the Random Tree in the RRT algorithm.

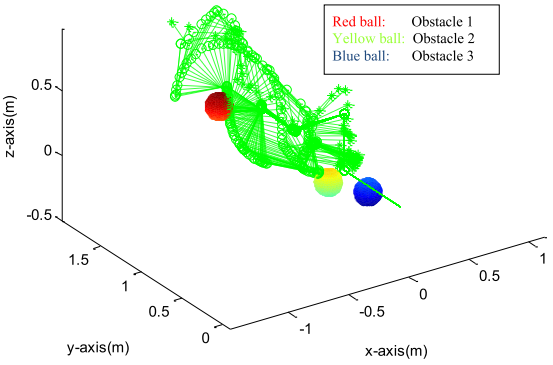


FIGURE 14. Planned path of the manipulator.

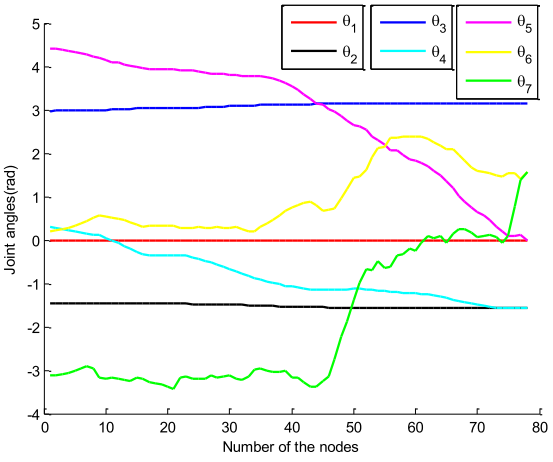


FIGURE 15. Planned paths of the joints angles of the manipulator.

the asterisks are all leaf nodes of the random tree in the RRT algorithm, that is, the exploring position for the end-effector of the manipulator; and the line is the planned path of the end-effector of the manipulator. These figures show the workspace explored by the obstacle avoidance and path planning algorithm. In FIGURE 10 and FIGURE 14, the asterisks are the positions of the end-effector at each path node, the circles are the joints of the manipulator at each path node, and the lines are the links of the manipulator at each node. These figures show the positions of the manipulator at each path node. In FIGURE 11 and FIGURE 15, the lines are the planned path of the joint angles of each joint. And, the label of the x-axis is Number of the nodes, it means the number of the iterations of the path planning algorithm. In this paper, the nodes are produced by the RRT algorithm, each iterative operative of the RRT algorithm will have a corresponding node in the search tree. therefore, the number of the nodes equal to the number of iterations. In FIGURE 12 and FIGURE 16, the lines show the planned path of positions of each joint.

FIGURE 9 shows that the RRT algorithm effectively explores the workspace, and finds a feasible path from the initial position to the target position for the end-effector of the manipulator. FIGURE 11 shows the path of the joint angles of each joint, and FIGURE 12 shows the path of each joint. When each joint moves along these paths, it ensures that the end-effector moves along the planned path shown in FIGURE 9. Comparing the planned path of each joint, it shows that the changes of joint 4, joint 5, joint 6, and joint 7 are much larger than the changes of joint 1, joint 2, and joint 3. FIGURE 10 shows that the manipulator can avoid the obstacles when each joint moves along the planned path.

FIGURE 13, FIGURE 14, FIGURE 15 and FIGURE 16 all show that the obstacle avoidance and path planning algorithm finds a feasible path from the initial position to the target position for the end-effector of the manipulator. Comparison of FIGURE 9 and FIGURE 13, show that the collision-free paths are planned in both cases. However, the leaf nodes in FIGURE 9 are far more than the leaf nodes in FIGURE 13. This shows that the searches speed of the RRT algorithm is greatly improved by the introduction of the method of generating a random point by artificial guidance. Comparisons between FIGURE 11 and 14, as well as FIGURE11 and FIGURE15, show that the paths that are planned in different cases have different trends. This is due to a large number of collision-free paths from the initial position to the target position, however, the RRT algorithm only finds one of them. However, comparison of FIGURE 11 and FIGURE 15, shows that the number of the path nodes in Case 1 is 80, and 76 in Case 2. That is to say, the improved algorithm enhances the search speed on the premise of finding a feasible path.

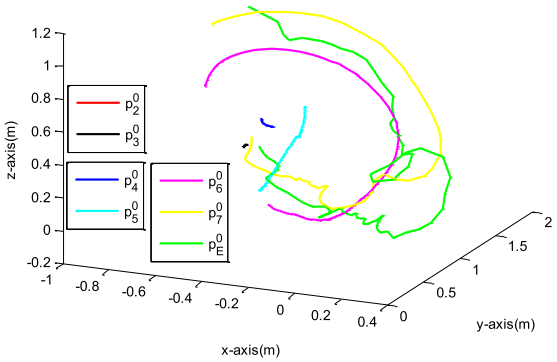


FIGURE 16. Planned paths of the positions of the joints of the manipulator.

In general, for the manipulator, the greater number of the joints, the higher maneuverability, and the greater performance of obstacle avoidance. That is, the obstacle avoidance function of the manipulator has requirement for the number of joints. In addition, the inverse kinematics problem of the manipulator in the space exist multiple solutions that only if the number of joints is greater than 3. FIGURE 11 and FIGURE 15 show that the joint angle changes in last four joints (joint 4, joint 5, joint 6, and joint 7) much larger than the joint angle changes in the first three joints (joint 1, joint 2, and joint 3). This shows that if the manipulator has four joints, it has the function of the obstacle avoidance. However, in the complex on-orbit environment, more joints of the manipulator are necessary for the space robot.

As a result, the FABRIK algorithm can calculate the inverse kinematics problem with low computational costs, since instead of using angle rotations, it treats finding the joint locations as a problem of finding a point on a line. And the obstacle avoidance and path planning algorithm can explore the workspace and quickly find a feasible obstacle-free path for the multi-joint manipulator, since the advantages of the RRT algorithm.

B. SIMULATION COMPARISON

Furthermore, in order to verify the advantage at low computational cost, some typical inverse kinematics algorithms are compared with the FABRIK algorithm, such as Jacobian Damped Least Squares algorithm (Jacobian DLS) [36], Cyclic Coordinate Descent algorithm (CCD) [16], Jacobian Transpose algorithm [37]. Therefore, several simulations and comparisons are implemented between the proposed algorithms respecting to the computational cost, and the runtime and the number of iterations have to reach the target. In these simulations, the inverse kinematics algorithms use the same initial parameters of the manipulator and the position of the target object, and the Jacobian and DLS parameters are the parameter values suggested by Buss, S, R. et, al. [37]. In addition, runtimes of the algorithms are measured with custom Matlab code on a personal computer (Intel(R) Core(TM) i7-4510U @ 2.00GHz).

TABLE 3 and TABLE 4 show the results of these simulations and comparisons. Here, the results are the average

TABLE 3. Average results of the FABRIK algorithm for the inverse kinematics problem.

Algorithm	Number of operations	Runtimes (sec)	Runtimes per operation (sec)
FABRIK	1	0.00047	0.00047
FABRIK	10	0.0045	0.00045
FABRIK	100	0.048	0.00048

TABLE 4. Average results of these typical algorithms for the inverse kinematics problem.

Algorithm	Number of iterations	Runtimes (sec)	Runtimes per operation (sec)
Jacobian DLS	925.748	6.3136	0.00682
CCD	28.213	0.0937	0.00317
Jacobian Transpose	1403.129	8.4749	0.00604

of 50 runs of the Matlab code. TABLE 3 shows the average runtimes of different operations using the FABRIK algorithm. And for these typical inverse kinematics algorithms, the number of iterations needed to reach the target object and the average runtimes are shown in TABLE 4.

TABLE 3 and TABLE 4 show that the runtimes of FABRIK algorithm are much lower than any other typical algorithms that mean the FABRIK algorithm has the advantage of low computational cost. An operation of FABRIK algorithm has the lowest computational cost, since instead of using angle rotations, it treats finding the joint locations as a problem of finding a point on a line.

V. CONCLUSION

This paper proposed a method applied to a multi-joint manipulator in the space robot to capture a target in a space environment with obstacles. The objective of the method is to avoid the obstacles by the path planning, principally, finding a feasible obstacle-free path from the initial position to the target position for the multi-joint manipulator. For this method, we present an improved obstacle avoidance and path planning algorithm applied to the multi-joint manipulator.

1) The improved algorithm is based on the RRT algorithm, which effectively explores the entire workspace. In the path planning algorithm, we use the position of the end-effector of the manipulator as the object of the planning, and use the obstacle avoidance as a key constraint.

2) To improve the search speed, we introduce the method of generating random point by artificial guidance.

3) Considering that the position of the end-effector depends on the joint angles of the joints of the manipulator, we design an improved method for the inverse kinematics problem to calculate the positions of the joints based on the improved FABRIK algorithm.

With the RRT algorithm and the improved FABRIK algorithm, the improved algorithm can plan a feasible path with low computational costs for the multi-joint manipulator in

the space environment with obstacles. The approach and algorithm proposed in this paper can ensure the safety of the movement of the multi-joint manipulator in a workspace with obstacles.

Future work will be devoted to developing an appropriate planning algorithm for quickly plan a feasible path for the space robot with multiple multi-joint manipulators in the space environment with obstacles. Also, it is worthy of verifying the proposed method in a real mobile robotics system.

REFERENCES

[1] A. Flores-Abad, O. Ma, K. Pham, and S. Ulrich, "A review of space robotics technologies for on-orbit servicing," *Prog. Aerosp. Sci.*, vol. 68, pp. 1–26, Jul. 2014.

[2] J. Vinals, E. Urgoiti, G. Guerra, I. Valiente, J. Esnoz-Larraya, M. Ilzkovitz, M. Franćeski, P. Letier, X.-T. Yan, G. Henry, A. Quaranta, W. Brinkmann, M. Jankovic, S. Bartsch, A. Fumagalli, and M. Doermer, "Multi–functional interface for flexibility and reconfigurability of future European space robotic systems," *Adv. Astronaut. Sci. Technol.*, vol. 1, no. 1, pp. 119–133, Sep. 2018.

[3] E. Falomir, S. Chaumette, and G. Guerrini, "Mobility strategies for swarms of unmanned aerial vehicles using artificial potential fields and global path planning," *Séminaire des Doctorants de la Société Informatique de France*, 2018, doi: 10.13140/RG.2.2.17741.33767.

[4] N. Bezzo, R. Fierro, A. Swingler, and S. Ferrari, "A disjunctive programming approach for motion planning of mobile router networks," *Int. J. Robot. Autom.*, vol. 26, no. 1, p. 13, 2011.

[5] B. Ranjbar, J. Mahmoodi, H. Karbasi, G. Dashti, and A. Omidvar, "Robot manipulator path planning based on intelligent multi-resolution potential field," *Int. J. u-e-Service, Sci. Technol.*, vol. 8, no. 1, pp. 11–26, 2015.

[6] V. Alekh, E. S. Rahul, and R. R. Bhavani, "Comparative study of potential field and sampling algorithms for manipulator obstacle avoidance," *Int. J. Control Theory Appl.*, vol. 9, pp. 71–78, 2016.

[7] H. Akbaripour and E. Masehian, "Semi-lazy probabilistic roadmap: A parameter-tuned, resilient and robust path planning method for manipulator robots," *Int. J. Adv. Manuf. Technol.*, vol. 89, nos. 5–8, pp. 1401–1430, Mar. 2017.

[8] K. Lochan, J. P. Singh, B. K. Roy, and B. Subudhi, "Hidden chaotic path planning and control of a two-link flexible robot manipulator," in *Nonlinear Dynamical Systems With Self-Excited and Hidden Attractors* (Studies in Systems, Decision and Control), vol. 133. 2018, pp. 433–463.

[9] Y. Nakamura and R. Mukherjee, "Nonholonomic path planning of space robots via a bidirectional approach," *IEEE Trans. Robot. Automat.*, vol. 7, no. 4, pp. 500–514, Aug. 1991.

[10] J. Hanxu and Z. Shuangqi, "Path planning for space manipulator to avoid obstacle based on A~* algorithm," *J. Mech. Eng.*, vol. 13, p. 17, 2010.

[11] Y. Xie, X. Wu, Z. Shi, Z. Wang, J. Sun, and T. Hao, "The path planning of space manipulator based on Gauss–Newton iteration method," *Adv. Mech. Eng.*, vol. 9, no. 8, Aug. 2017, Art. no. 168781401771847.

[12] M. A. Torres and S. Dubowsky, "Minimizing spacecraft attitude disturbances in space manipulator systems," *J. Guid., Control, Dyn.*, vol. 15, no. 4, pp. 1010–1017, Jul. 1992.

[13] K. Yamada and S. Yoshikawa, "Feedback control of space robot attitude by cyclic arm motion," *J. Guid., Control, Dyn.*, vol. 20, no. 4, pp. 715–720, Jul. 1997.

[14] S. Cocuzza, I. Pretto, and S. Debei, "Least-squares-based reaction control of space manipulators," *J. Guid., Control, Dyn*, vol. 35, no. 3, pp. 976–986, May 2012.

[15] P. Huang, Y. Xu, and B. Liang, "Dynamic balance control of multi-arm free-floating space robots," *Int. J. Adv. Robot. Syst.*, vol. 2, no. 2, pp. 398–403, 2006.

[16] L.-C. Wang and C. Chen, "A combined optimization method for solving the inverse kinematics problems of mechanical manipulators," *IEEE Trans. Robot. Automat.*, vol. 7, no. 4, pp. 489–499, Aug. 1991.

[17] S. R. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE J. Robot. Automat.*, vol. 17, no. 16, pp. 1–19, Apr. 2004.

[18] S. Qiao, Q. Liao, S. Wei, and H.-J. Su, "Inverse kinematic analysis of the general 6R serial manipulators based on double quaternions," *Mechanism Mach. Theory*, vol. 45, no. 2, pp. 193–199, Feb. 2010.

[19] N. Rokbani and A. Alimi, "Inverse kinematics using particle swarm optimization, a statistical analysis," *Procedia Eng.*, vol. 64, pp. 1602–1611, Jan. 2013.

[20] Y. Wei, S. Jian, S. He, and Z. Wang, "General approach for inverse kinematics of nR robots," *Mechanism Mach. Theory*, vol. 75, pp. 97–106, May 2014.

[21] A.-V. Duka, "ANFIS based solution to the inverse kinematics of a 3DOF planar manipulator," *Procedia Technol.*, vol. 19, pp. 526–533, Jan. 2015.

[22] K. L. Doty, C. Melchiorri, and C. Bonivento, "A theory of generalized inverses applied to robotics," *Int. J. Robot. Res.*, vol. 12, no. 1, pp. 1–19, Feb. 1993.

[23] L. Sciavicco and B. Siciliano, "Modelling and control of robot manipulators," *Meas. Sci. Technol.*, vol. 11, no. 12, pp. 21–29, 2012.

[24] S. Yahya, M. Moghavvemi, and H. A. Mohamed, "Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace," *Simul. Model. Pract. Theory*, vol. 19, no. 1, pp. 406–422, Jan. 2011.

[25] S. C. Wang, *Interdisciplinary Computing in Java Programming* (The Springer International Series in Engineering and Computer Science), vol. 743. 2012, pp. 81–100.

[26] R. Singh, V. Vishal, T. N. Singh, and P. G. Ranjith, "A comparative study of generalized regression neural network approach and adaptive neuro-fuzzy inference systems for prediction of unconfined compressive strength of rocks," *Neural Comput. Appl.*, vol. 23, no. 2, pp. 499–506, Aug. 2013.

[27] M. Vande Weghe, D. Ferguson, and S. S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," in *Proc. 7th IEEE-RAS Int. Conf. Hum. Robots*, Nov. 2007.

[28] A. Sintov and A. Shapiro, "Time-based RRT algorithm for rendezvous planning of two dynamic systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014.

[29] T. Rybus and K. Seweryn, "Application of rapidly-exploring random trees (RRT) algorithm for trajectory planning of free-floating space manipulator," in *Proc. 10th Int. Workshop Robot Motion Control (RoMoCo)*, Jul. 2015.

[30] J. R. Benevides and V. Grassi, "Autonomous path planning of free–floating manipulators using RRT–based algorithms," in *Proc. 12th Latin Amer. Robot. Symp. 3rd Brazilian Symp. Robot. (LARS-SBR)*, Oct. 2015.

[31] L. Radavelli, R. Simoni, P. E. De, and D. Martins, "A comparative study of the kinematics of robots manipulators by Denavit-Hartenberg and dual quaternion," *Mecánica Comput., Multi-Body Syst.*, vol. 31, no. 15, pp. 2833–2848, 2012.

[32] M. Wang, J. Luo, J. Yuan, and U. Walter, "Coordinated trajectory planning of dual-arm space robot using constrained particle swarm optimization," *Acta Astronaut.*, vol. 146, pp. 259–272, May 2018.

[33] H.-C. Lee, T. Yaniss, and B.-H. Lee, "Grafting: A path replanning technique for rapidly-exploring random trees in dynamic environments," *Adv. Robot.*, vol. 26, no. 18, pp. 2145–2168, 2012.

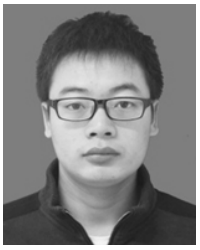
[34] S. M. Lavalle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001.

[35] A. Aristidou and J. Lasenby, "FABRIK: A fast, iterative solver for the Inverse Kinematics problem," *Graph. Models*, vol. 73, no. 5, pp. 243–260, Sep. 2011.

[36] C. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least–squares methods," *IEEE Trans. Syst., Man, Cybern.*, vol. 16, no. 1, pp. 93–101, Jan. 1986.

[37] S. R. Buss and J.-S. Kim, "Selectively damped least squares for inverse kinematics," *J. Graph. Tools*, vol. 10, no. 3, pp. 37–49, Jan. 2005.

[38] S. Xu, H. Wang, D. Zhang, and B. Yang, "Adaptive reactionless motion control for free-floating space manipulators with uncertain kinematics and dynamics," in *Proc. 3rd IFAC Int. Conf. Intell. Control Automat. Sci.*, vol. 46, 2013, pp. 646–653.



YAEN XIE received the B.S. degree in information and computing science from Liaoning University, Shenyang, China, in 2013, and the M.S. degree in aerospace engineering from Harbin Engineering University, Harbin, China, in 2015, where he is currently pursuing the Ph.D. degree in control science and engineering.

From 2018 to 2019, he was a Special Research Student with the Graduate School of Engineering, Nagoya University. His research interests include the dynamics of the complex spacecraft systems and the dynamics and control of the spacecraft, and dynamics and control of the spatial multibody systems.

ZHIDAN ZHANG was born in Shangcai, China, in 1994. She received the B.S. degree in information and computing science from Zhengzhou University, Zhengzhou, China, in 2017. She is currently pursuing the M.S. degree in aerospace engineering with Harbin Engineering University, Harbin, China.

Her research interests include the reliability of the integrated electronic systems and the dynamics and control of the spacecraft, and dynamics and control of the spatial multibody systems.

XIANDE WU received the bachelor's degree from Central South University, in 2002, and the Ph.D. degree from the Harbin Institute of Technology, in 2010. He is currently an Associate Professor with the College of Aerospace and Civil Engineering, Harbin Engineering University. His research interests include dynamics and control, system modeling and simulation, and path and task planning.

ZHEN SHI received the B.S. degree in aerospace engineering and the M.S. degree in control engineering from the Institute of Harbin Shipbuilding Engineering, Harbin, China, in 1982 and 1986, respectively, and the Ph.D. degree in control engineering from Harbin Engineering University, Harbin, in 2001.

From 1996 to 2001, he was an Assistant Professor with the Institute of Harbin Shipbuilding Engineering. Since 2001, he has been a Professor with the Harbin Engineering University. His research interests include the guidance and control technology for aerocraft, the control theory and its application, and the strap-down inertial navigation technology.

YANGYANG CHEN received the B.S. and M.S. degrees from Harbin Engineering University, Harbin, China, in 2010 and 2013, respectively.

He has been an Engineer with the Beijing Institute of Electronic System Engineering. His research interests include missile design and trajectory simulation.

BAIXUAN WU was born in Lianyuan, China, in 1995. He received the B.S. degrees in automation from the Harbin Institute of Technology, Weihai, China, in 2017. He is currently pursuing the M.S. degree in aerospace engineering with Harbin Engineering University, Harbin, China.

His research interests include the reliability of the integrated electronic systems and the dynamics and control of the spacecraft, and dynamics and control of the spatial multibody systems.

KOFI AKROFI MANTEY received the B.S. degree in building technology from the Kwame Nkrumah University of Science and Technology, Kumasi, Ghana, in 2006, and the M.Eng. degree in civil engineering from Harbin Engineering University, Habrin, China, in 2019. He worked in the construction industry, initially as a Site Engineer and subsequently as an Operations Manager, undertaking the construction of 763 affordable homes. His research interests include ultrahigh performance concretes, prestressing technology, and the development of computer-aided engineering software.