Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Spline-RRT*: Coordinated Motion Planning of Dual-Arm Space Robot ⋆

Min Yu, * **Jianjin Luo, Mingming Wang, Dengwei Gao**

* *Research & Development Institute of Northwestern Polytechnical University in Shenzhen, Shenzhen, CO 518057 China (e-mail: yumin@mail.nwpu.edu.cn).*

Abstract: This paper addresses the coordinated motion planning issue for a dual-arm free-floating space robot. Based on the sampling-based planning method, a novel coordinated RRT*-based path planning framework is proposed for a dual-arm manipulation. First, an asymptotically optimal sampling-based method, RRT*, is employed to generate the initial rough path for each end-effector in the task-space. To avoid self-collision, we present a coordinated strategy which samples from separated inertial spaces when performing RRT* algorithm. Second, quartic splines are used to smooth the generated RRT* path so that the robot can execute smoothly. Physical constraints including the end-effectors' limit, joint limit and boundary conditions are all controlled within the design of the quartic splines. The effectiveness of the proposed path planning framework is illustrated and demonstrated via a kinematically redundant dual-arm space robot.

Keywords: Space robot, coordinated path planning, RRT*, quartic splines, motion control.

1. INTRODUCTION

Robotic technology applied in an unstructured space environment has been a constant hit recently. The successes of series of experiments by applying space robot, such as Engineering Test Satellite VII (ETS-VII) and Front-End Robotics Enabling Near-Term Demonstration (FREND), indicate that robot is a prospective tool for on-orbit servicing mission.

As one of the most challenging problems among robotic technologies, motion planning has received great attentions in the past. Traditionally, Nenchev (1992) used reaction null space to obtain the best robot's configuration. A point-to-point planning method was proposed in Xu (2009) to synchronously plan the base attitude as well as the pose of end-effector. Afterwards, Aghili (2013) studied the motion planning issue when capturing a tumbling target for both pre- and post-grasping phases. Lampariello (2014) calculated the joint trajectories for space robot within a useful time. See Wang (2018), Bézier curve and Particle Swarm Optimization method were employed for the coordinated path planning and trajectory planning, respectively. Abad (2014) designed an optimal control scheme, in which the uncertainties in the initial and final boundary conditions were considered. Nevertheless, the aforementioned literatures seldom address the path generation issue for the motion of space robot in an unstructured environment.

Recently, the sampling-based method, such as probabilistic roadmap (PRM), rapidly exploring random trees (RRT) and their variants, presents a novel motion planning approach particularly in an unstructured environment, see LaValle (2006), and it has shown a great, universal capability for a wide range of applications, including many in the robot field. Persson (2014) proposed a sampling-based A* algorithm for path planing of a 7 degree-of-freedom space manipulator. James (2016) employed RRT algorithm to plan motions from a given state to a goal state for a dual-arm space robot. Vahrenkamp (2010) proposed a simultaneous grasp and motion planning strategy for a humanoid robot via RRT-based algorithms. Webb (2013) presented kinodynamic RRT* for asymptotically optimal motion planning for robots with linear dynamics. Rybus (2014) employed RRT algorithm to minimize rotational kinetic energy and stabilize motion. Although sampling-based method is efficient and probabilistically complete for high-dimensional spaces, the resulting path is often jerky and thus requires a smoothing technique for the robot motion planning.

Currently existing smoothing techniques fall into two categories: optimization-based methods and curve-fitting methods, see Zhu (2015). Common optimization techniques, such as gradient-based methods and convex optimization, could handle dynamic constraints explicitly, but it takes rather seconds to obtain a feasible smooth trajectory. In contrast, curve-fitting techniques replace the jerky portions with curve segments, such as splines, which are rather fast. A hierarchical planning framework, path planning with path parameterization, for robots is validated to be efficient. Ding (2019) studied the trajectory planning issue for quadrotors under the kinodynamic planning framework and B-splines was employed to guarantee the safety and dynamic feasibility. Salaris (2019) also adopted B-splines to design the trajectory of a

⋆ This research was supported by Science, Technology and Innovation Commission of Shenzhen Municipality (Grant No. JCYJ20180508151938535) and The National Natural Science Foundation of China (Grant No. 61973256, 61690210 and 61690211).

unicycle and a quadrotor respectively. Vaz (2006) planned the robot trajectory with cubic splines and formulated the planning problem as a standard semi-infinite programming issue. Since the use of smooth, parameterized splines has been proven particularly effective in the robot field, see Kolter (2009), quartic splines are employed in this paper to smooth the jerky portions of a path.

Herein, we present a sampling-based path planning framework incorporating with quartic splines for a coordinated dual-arm manipulation, which can be referred to as a two-phase approach: path planning and spline fitting. Firstly, we use a high-level planning algorithm, RRT* algorithm, to generate a series of kinematically feasible waypoints that the robot should pass through on its way to the goal. Note that the coordination of the dual-arm manipulation is also considered to avoid possible self-collisions by separating the sampling space in the off-the-shelf RRT* algorithm. Secondly, we fit the parameters of quartic splines passing through the generated waypoints. Cubic splines are ubiquitous in the field of robot planning, while quartic splines are rarely employed. Nonetheless, quartic splines are necessary to satisfy the third derivative continuity (particularly for the robot motion) of the planned path, as well as boundary constraints, the first derivative continuity and the second derivative continuity. The proposed path planning framework is evaluated on a kinematically redundant dual-arm space robot.

2. MODELING

A free-floating dual-arm space robot consists of a spacecraft and two n-degree-of-freedom manipulators under no external forces and no base actuation. Hypothesize that the spacecraft and manipulators' links can be regarded as rigid bodies. The kinematics of the end-effectors is

$$\begin{bmatrix} \dot{\boldsymbol{x}}_e^a \\ \dot{\boldsymbol{x}}_e^b \end{bmatrix} = \begin{bmatrix} \mathbf{J}_b^a \dot{\boldsymbol{x}}_b + \mathbf{J}_e^a \dot{\boldsymbol{\theta}}^a \\ \mathbf{J}_b^b \dot{\boldsymbol{x}}_b + \mathbf{J}_e^b \dot{\boldsymbol{\theta}}^b \end{bmatrix} \tag{1}$$

Superscripts a and b represent that they are respectively related to the manipulator a and b, and subscripts b and e denotes the base spacecraft and end-effectors, respectively. $\dot{\boldsymbol{x}}_b, \dot{\boldsymbol{x}}_e^a, \dot{\boldsymbol{x}}_e^b$ denote velocities of the base and end-effectors. $\dot{\boldsymbol{\theta}}$ denotes the joint velocity.

Based on Lagrangian mechanism principle, the dynamics of the system is

$$\begin{bmatrix} \mathbf{H}_b & \mathbf{H}_{bm}^a & \mathbf{H}_{bm}^b \\ \mathbf{H}_{bm}^{a\mathrm{T}} & \mathbf{H}_m^a & \mathbf{0}_n \\ \mathbf{H}_{bm}^{b\mathrm{T}} & \mathbf{0}_n & \mathbf{H}_m^b \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{x}}_b \\ \ddot{\boldsymbol{\theta}}^a \\ \ddot{\boldsymbol{\theta}}^b \end{bmatrix} + \begin{bmatrix} \boldsymbol{c}_b \\ \boldsymbol{c}_m^a \\ \boldsymbol{c}_m^b \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_b \\ \boldsymbol{\tau}_m^a \\ \boldsymbol{\tau}_m^b \end{bmatrix} \tag{2}$$

Since there is no external forces or torques imposed on the whole system in a free-floating type, i.e., $\boldsymbol{f}_b = \mathbf{0}$. Based on the momentum conservation principle, a more concise kinematics equation can be written as

$$\begin{bmatrix} \boldsymbol{P}_0 \\ \boldsymbol{L}_0 \end{bmatrix} = \mathbf{H}_b \begin{bmatrix} \dot{\boldsymbol{r}}_b \\ \boldsymbol{\omega}_b \end{bmatrix} + \mathbf{H}_{bm}^a \dot{\boldsymbol{\theta}}^a + \mathbf{H}_{bm}^b \dot{\boldsymbol{\theta}}^b \tag{3}$$

where $\boldsymbol{M}_0 = [\boldsymbol{P}_0, \boldsymbol{L}_0]^{\mathrm{T}}$ denote the initial linear and angular momentums of the system. $\boldsymbol{H}_b$ is invertible, the motion of the base can be expressed by

$$\begin{bmatrix} \dot{\boldsymbol{r}}_b \\ \boldsymbol{\omega}_b \end{bmatrix} = \begin{bmatrix} -\mathbf{H}_b{}^{-1}\mathbf{H}_{bm}^a & -\mathbf{H}_b{}^{-1}\mathbf{H}_{bm}^b \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\theta}}^a \\ \dot{\boldsymbol{\theta}}^b \end{bmatrix} - \mathbf{H}_b{}^{-1}\boldsymbol{M}_0 \tag{4}$$

The motion of the end-effectors is finally given as

$$\begin{bmatrix} \mathbf{J}_g^a \\ \mathbf{J}_g^b \end{bmatrix} \dot{\boldsymbol{\theta}} = \begin{bmatrix} \mathbf{J}_e^a - \mathbf{J}_b^a \mathbf{H}_b{}^{-1}\mathbf{H}_{bm}^a & -\mathbf{J}_b^a \mathbf{H}_b{}^{-1}\mathbf{H}_{bm}^b \\ -\mathbf{J}_b^b \mathbf{H}_b{}^{-1}\mathbf{H}_{bm}^a & \mathbf{J}_e^b - \mathbf{J}_b^b \mathbf{H}_b{}^{-1}\mathbf{H}_{bm}^b \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\theta}}^a \\ \dot{\boldsymbol{\theta}}^b \end{bmatrix} \tag{5}$$

where $\mathbf{J}_g$ is termed Generalized Jacobian Matrix.

3. COORDINATED MOTION PLANNING

In this section, a novel spline-RRT* framework is proposed for the coordinated motion of a dual-arm space robot. The overview and a compact algorithm of the presented framework are given in Fig. 1 and Table 1 respectively. An initial rough path for the dual-arm system is obtained by improving the off-the-shelf RRT* planning algorithm, which samples from two separated inertial spaces to avoid possible self-collisions. Then, quartic splines reparameterize and smooth the generated RRT* path so that the robot can execute it, in which continuous velocity, acceleration and jerk profiles are all considered. The presented sampling-based motion planning framework can efficiently find a kinematically feasible path for the dual-arm space robot.

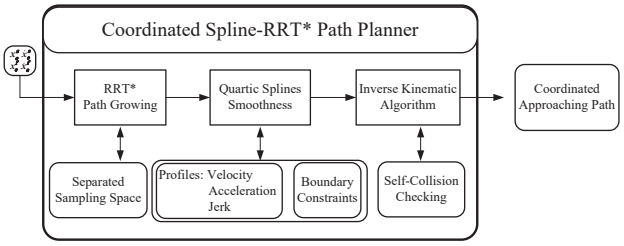

Fig. 1. Overview of coordinated spline-RRT* planner

Table 1. Coordinated spline-RRT* algorithm

1: Given end-effectors' initial state and desired final state
2: *Coordinated-RRT**
3: **For** $k = 1$**to**K **do**
4: Sampling from *separated* task spaces
5: ExtendRandomly, CollisionCheck
6: **Return** Waypoints(RRT*)
7: *Quartic Splines Smoothness*
8: **Profile** Waypoints(RRT*) with quartic splines
9: Compute **parameters** of quartic splines
10: **Fit** splines with Waypoints(RRT*)
11:**Return** Path(spline-RRT*)

3.1 Task-space path planning

The end-effectors' path $\boldsymbol{x}(t)$ is generated via two decoupled methods, RRT* for the position and a Bézier curve for the attitude. A fifth-order Bézier curve is employed to plan the desired path in our early work (Luo (2018)). Thus, it is omitted here for brevity reason. We mainly focus on the coordinated path planned for the end-effectors' position via RRT* here.

Given a desired end-effectors' task in the task space, the path planing problem can be formulated as seeking for a feasible end-effectors' path that satisfies boundary constraints and physical constraints such as end-effectors' limits while solving for the inverse kinematic of the robot. For an end-effector's task, its initial conditions (position

$\boldsymbol{x}_e^s$, velocity $\dot{\boldsymbol{x}}_e^s$ and acceleration $\ddot{\boldsymbol{x}}_e^s$) and final conditions (position $\boldsymbol{x}_e^f$ and velocity $\dot{\boldsymbol{x}}_e^f$) can be determined beforehand. The initial acceleration is set to zeros for minimizing joint jerk, but there is no restriction on the final acceleration herein. The path planning task of optimization the location of waypoints while obeying certain constraints can be stated as follows:

$$\begin{aligned}
&\min : f(\boldsymbol{x}_e) \\
&\text{s.t.} : \boldsymbol{x}_e(t=0) = \boldsymbol{x}_e^s, \boldsymbol{x}_e(t=t_f) = \boldsymbol{x}_e^f \\
&\quad \dot{\boldsymbol{x}}_e(t=0) = \dot{\boldsymbol{x}}_e^s, \dot{\boldsymbol{x}}_e(t=t_f) = \dot{\boldsymbol{x}}_e^f \\
&\quad \ddot{\boldsymbol{x}}_e(t=0) = \ddot{\boldsymbol{x}}_e^s \\
&\quad \boldsymbol{\theta}(t=0) = \boldsymbol{\theta}^s, \dot{\boldsymbol{\theta}}(t=0) = \dot{\boldsymbol{\theta}}^s, \ddot{\boldsymbol{\theta}}(t=0) = \ddot{\boldsymbol{\theta}}^s \\
&\quad \dot{\boldsymbol{\theta}}_{min} \le \dot{\boldsymbol{\theta}} \le \dot{\boldsymbol{\theta}}_{max}, \ddot{\boldsymbol{\theta}}_{min} \le \ddot{\boldsymbol{\theta}} \le \ddot{\boldsymbol{\theta}}_{max}
\end{aligned} \tag{6}$$

where $\boldsymbol{x}_e \in \mathbb{R}^3$ is the end-effector's position. $f(\boldsymbol{x}_e)$ represents the location of the waypoints. $\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \ddot{\boldsymbol{\theta}}$ are joint angle, velocity and acceleration respectively, and $\boldsymbol{\theta}_{min}, \boldsymbol{\theta}_{max}, \dot{\boldsymbol{\theta}}_{min}, \dot{\boldsymbol{\theta}}_{max}, \ddot{\boldsymbol{\theta}}_{min}, \ddot{\boldsymbol{\theta}}_{max}$ are the corresponding joint limits.

3.2 Coordinated RRT path planning*

Explicit details about the RRT* can be found in Karaman (2011). To seek a path from a given initial state to a final state of end-effectors, we use the RRT* algorithm to create a tree in the task-space. Work of the RRT* algorithm for our path planning issue can be expressed as follows. Firstly, an initial position of end-effectors is inserted as a first vertex of a tree and then a random position point is sampled from the task-space. Next, a tree vertex nearest to this point is located and a collision check is performed to ensure the position of the end-effectors is safe. After that, a new vertex is obtained and then a rewiring operation is executed to check if there is any other minimum-cost path. The procedure is repeated until the final position is reached.

Since we aim at planning for a dual-arm space robot, there are two paths generated by the RRT* algorithm for each end-effector. It is quite possible to get stuck into self-collisions due to the random samplings of the RRT* algorithm. If we check self-collisions during the growing procedure, it would be time-consuming. Therefore, we proposed a novel coordinated RRT* algorithm by randomly sampling from two separated task-spaces for each end-effector. That is to say, imposing a virtually wall-like space constraint between the end-effectors when planning to ensure no self-collisions. Moreover, this wall-like space constraint should not be a constant one but varies along with the motion of the end-effectors, or termed as a continuously varying space constraint. This coordinated strategy is also universal for the path planning issue of other multi-agent systems.

Directly connecting waypoints generated by RRT* leads to an extremely rough path, which is harmful to the robot execution. As a result, splines, passing though all these waypoints, incorporated with the coordinated RRT* algorithm enables a smoother robot motion.

3.3 Quartic splines smoothing technique

One strategy that has been proven particularly effective for smoothing paths is the use of smooth, parametrized splines to describe paths. First of all, let us recall procedures for fitting quartic splines to waypoints planned by RRT* algorithm. The waypoints can be expressed as the following time-location pairs.

$$(t_0, x_0), (t_1, x_1), \cdots, (t_n, x_n) \tag{7}$$

where x_i is the planned position of the end-effector at time t_i.

Once these waypoints are predetermined, a unique piecewise quartic trajectory passes through these points and satisfies certain smoothness criteria. Considering the trajectory between times t_i and t_{i+1}, denoted as $x_i(t) : \mathbb{R} \to \mathbb{R}^n$, as a quartic function

$$x_i(t) = a_i + b_i(t-t_i) + c_i(t-t_i)^2 + d_i(t-t_i)^3 + e_i(t-t_i)^4 \tag{8}$$

where $a_i, b_i, c_i, d_i, e_i \in \mathbb{R}^n$ are parameters of the quartic spline. The final trajectory $x(t) : \mathbb{R} \to \mathbb{R}^n$ is a piecewise quartic function that is simply the concatenation of these different quartic trajectories.

$$x(t) = \begin{cases} x_0(t), & \text{if } t_0 < t \le t_1 \\ \vdots & \\ x_{T-1}(t), & \text{if } t_{T-1} < t \le t_T \end{cases} \tag{9}$$

For the robot motion, jerk profiles at each waypoint should also be continuous. These profiles can be satisfied via the first-order, second-order and third-order derivatives of the quartic splines. Thereafter, we can calculate the parameters of the quartic splines by solving the functions and derivatives with the given waypoints.

$$\begin{aligned}
x_i'(t) &= b_i + 2c_i(t-t_i) + 3d_i(t-t_i)^2 + 4e_i(t-t_i)^3 \\
x_i''(t) &= 2c_i + 6d_i(t-t_i) + 12e_i(t-t_i)^2 \\
x_i'''(t) &= 6d_i + 24e_i(t-t_i)
\end{aligned} \tag{10}$$

Taking a simple case for example, a sketch of the quartic splines is given in Fig. 2. Supposing that there are 4 waypoints in one end-effector's path of the robot. The second and the third waypoints are defined as joint waypoints. In this case, there should be 3 pieces of quartic trajectories and 15 parameters (5 parameters for each piece) which corresponds to 15 unknown equations needed solving.

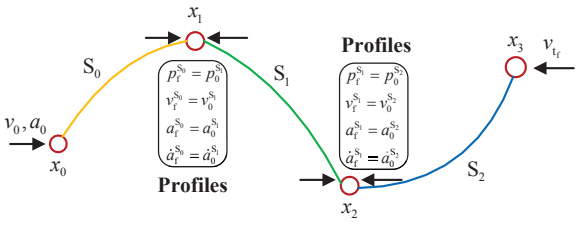

Fig. 2. A sketch of the quartic splines

There are 4 position constraints for the 4 desired waypoints as well as 8 constraints for the continuous position, velocity, acceleration and jerk profiles at two joint waypoints, x_1 and x_2. For the robot motion, the initial velocity, acceleration and the final velocity of the end-effector should be constrained. As mentioned earlier, there is no need to

restrain the final acceleration here. Totally, there are 15 constraints for constructing the equations to solve the 15 unknown parameters of quartic splines. Finally, the linear equations with a size of 15 of the proposed case can be stated as follows.

$$\begin{cases} a_0 = x_0, a_1 = x_1, a_2 = x_2, \\ a_0 + b_0 h_0 + c_0 h_0^2 + d_0 h_0^3 + e_0 h_0^4 = x_1, \\ a_1 + b_1 h_1 + c_1 h_1^2 + d_1 h_1^3 + e_1 h_1^4 = x_2, \\ a_2 + b_2 h_2 + c_2 h_2^2 + d_2 h_2^3 + e_2 h_2^4 = x_3, \\ b_0 + 2c_0 h_0 + 3d_0 h_0^2 + 4e_0 h_0^3 - b_1 = 0, \\ b_1 + 2c_1 h_1 + 3d_1 h_1^2 + 4e_1 h_1^3 - b_2 = 0, \\ c_0 + 3d_0 h_0 + 6e_0 h_0^2 - c_1 = 0, \\ c_1 + 3d_1 h_1 + 6e_1 h_1^2 - c_2 = 0, \\ d_0 + 4e_0 h_0 - d_1 = 0, \\ d_1 + 4e_1 h_1 - d_2 = 0, \\ S_0'(x_0) = b_0 = init\ vel, S_0''(x_0) = 2c_0 = init\ acc, \\ S_2'(x_3) = b_2 + 2c_2 h_2 + 3d_2 h_2^2 + 4e_2 h_2^3 = final\ vel. \end{cases} \tag{11}$$

where h_0, h_1, h_2 depend only on the times t_0, t_1, t_2, t_3. The former 6 equations are the position profiles for each waypoint, while the middle 6 equations are the velocity, acceleration and jerk profiles for each joint waypoint. Besides, the last 3 equations are the initial velocity, initial acceleration and final velocity constraints of the end-effectors. Thus, a complete definition of the parameters of the quartic splines for the given case is presented and the parameters can be easily solved by the linear equations in the desired locations.

Note that the coordinated RRT* algorithm is used to find feasible wayppoints in advance, hence the computational complexity main stems from quartic splines, i.e., the linear equation constraints, which are easily solved and make this framework viable for a fast planning implementation.

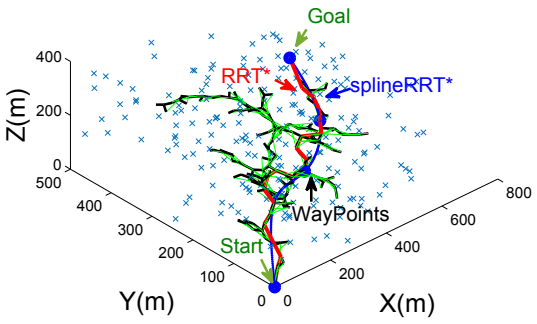

Fig. 3. Simulation result of spline-RRT* algorithm

Fig. 3 gives the simulation result of our proposed spline-RRT* algorithm. The initial position is the origin of the coordinate frame and the final position is $(640, 480, 400)^{\mathrm{T}} m$. A normalized time interval, $[0, 1]$, is used and split into three uniform subintervals for 4 waypoints. The blue crosses denote the samplings. The black lines are the branches of the growing tree, while the slim, green lines are the rewired branches. The red zigzag curve is the initial path planned by RRT* algorithm, whereas the blue curve is the final path generated by the polished spline-RRT* algorithm. Obviously, our proposed spline-RRT* algorithm could plan a feasible, smooth path for the robot motion, as well as the optimality of the path is preserved.

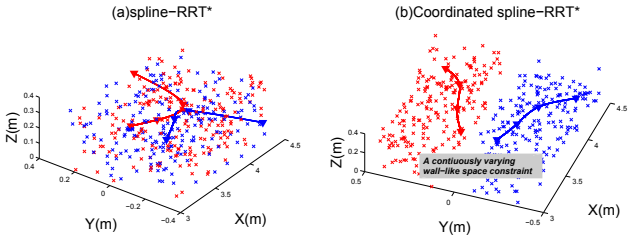

Fig. 4. Illustration of coordinated spline-RRT* algorithm

Moreover, we also conducted a simulation case to illustrate the coordination of our proposed spline-RRT* algorithm. As we can see, if randomly sampling from the inertial space, the end-effectors may collide with each other as shown in Fig. 4(a), whereas the case is drastically refined when sampling from the separated inertial spaces as shown in Fig. 4(b).

4. SIMULATION RESULTS

The proposed coordinated spline-RRT* framework is numerically validated using a continuous end-effectors' tracking example. The goal of the planning algorithm is to track an approaching path while adhering to the end-effectors' limit and joint limit, as well as ensuring a collision-free motion. The initial pose is

$$x_e^{as} = (3.556, -0.101, 0.168, -\frac{\pi}{2}, 0, \frac{\pi}{4}) \tag{12}$$

$$x_e^{bs} = (3.556, 0.101, 0.168, \frac{\pi}{2}, 0, -\frac{\pi}{4}) \tag{13}$$

and the final pose is

$$x_e^{af} = (4.271, -0.365, 0.168, -\frac{\pi}{2}, 0, \frac{\pi}{6}) \tag{14}$$

$$x_e^{bf} = (4.271, 0.365, 0.168, \frac{\pi}{2}, 0, -\frac{\pi}{6}) \tag{15}$$

The total time T of operation is calculated by considering the end-effectors' limit:

$$T \geq max(\frac{\|\dot{\boldsymbol{x}}_{max}\|}{\dot{\boldsymbol{x}}_{max}}, \sqrt{\frac{\|\ddot{\boldsymbol{x}}_{max}\|}{\ddot{\boldsymbol{x}}_{max}}}) \tag{16}$$

Once the execution time T is determined, the end-effectors' path in the task-space using the spline-RRT* can be generated properly.

Fig. 5 illustrates the planned path by the coordinated spline-RRT* algorithm for the end-effectors. As we can see from Fig. 5(a) and (b), the path planned by our proposed algorithm varies smoothly from the given initial pose to the goal pose for both arms. The velocity of the end-effectors is restricted within the end-effectors' physical limit, 0.1m/s, as shown in Fig. 5(c) and (d). More importantly, the initial and final velocity of the end-effectors are zeros, which matches our original settings in splines fitting, and the initial acceleration of the end-effectors is also zero from Fig. 5(c) and (d). Considering the path velocity and acceleration boundaries, the execution time T is

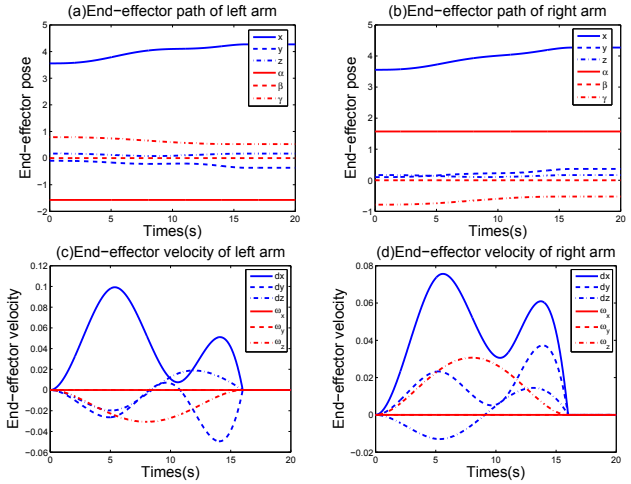

Fig. 5. Planned path

calculated, where $T = 16s$. Therefore, the designed end-effectors' path fulfils the imposed constraints and required tracking mission in task-space.

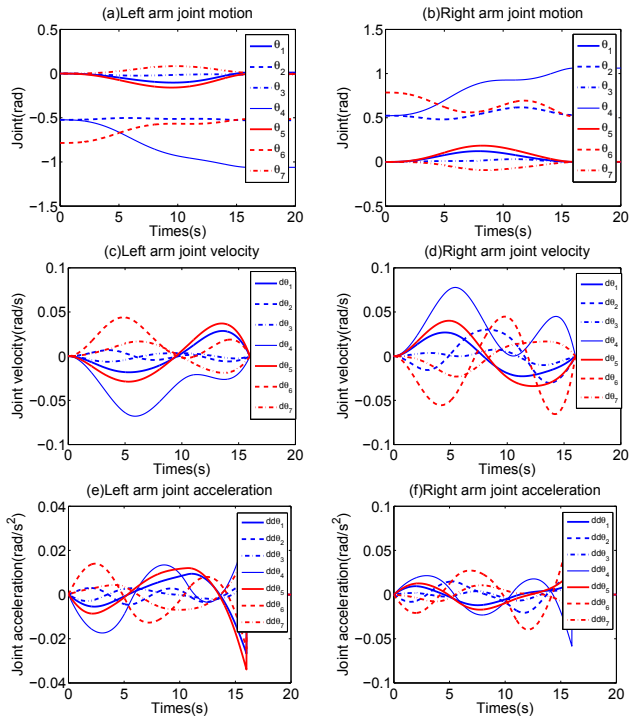

Fig. 6. Joint motion, velocity and acceleration

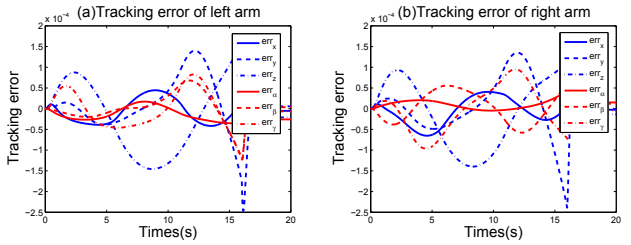

Fig. 7. Tracking error of the end-effectors

The corresponding robot executive outcomes are given in Fig. 6 and Fig. 7. When executing the designed path, the joint trajectories are smooth and within the joint limits, which are $0.1deg/s$ for joint velocity and $0.1deg/s^2$ for joint acceleration. Moreover, the initial joint velocity and acceleration are restricted to zeros for minimizing the joint jerk. The order of magnitude of the tracking position and orientation errors for both end-effectors is as lower as 10^{-4} as shown in Fig. 7. Therefore, the tracking task of the end-effectors is well done with the generated joint trajectories.

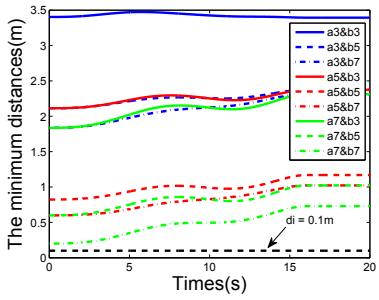

Fig. 8. The minimum distances between robot links

Fig. 8 gives the minimum distances between the robot links. We only detect the minimum distances between main robot links, link 3, 5 and 7 for each arm, which are likely to collide mutually. Thanks to our separated sampling strategy, there is no any collision during the robot execution. The minimum distances are always larger than the influence distance, $d_i = 0.1m$. Fig. 8 further validate the coordination of our proposed algorithm.

Moreover, we conduct the simulation case both in the MATLAB/Simulink environment and Virtual Reality (VR) platform at the same time. The results of the Virtual Reality are given in Fig. 9, demonstrating the correctness of our algorithm. When catching a moving object, the manipulators unfold at first. By using our proposed motion planning algorithm, the pose of the end-effectors approaches the object gradually. The RGB (R for red, G for green and B for blue) coordinate frames in X-Y-Z order we use for all frames. Finally, it's clear that the coordinate frames of both end-effectors matches the counterpart of the object, which illustrates our algorithm successfully tracks the moving object.

5. CONCLUSION

A coordinated motion planning scheme for a free-floating dual-arm space robot is presented in this paper. This approach uses an optimal, rapidly exploring random trees (RRT*) algorithm, as a high-level planning phase, to seek reasonable waypoints for end-effectors. A novel coordinated strategy, sampling from separated inertial spaces for each end-effector, is proposed for the sake of possible self-collisions. Then, quartic splines are employed to concatenate the desired waypoints generated via coordinated RRT* algorithm and eventually obtain a smooth motion for the robot, in which end-effectors' physical limit, joint limit and boundary constraints are all taken into consideration. Simulation results validate the effectiveness and correctness of the proposed spline-RRT* algorithm. The presented coordinated motion planning scheme can also be applied to the path planning issue for ground robots and multi-agent systems, in which a more well-thought

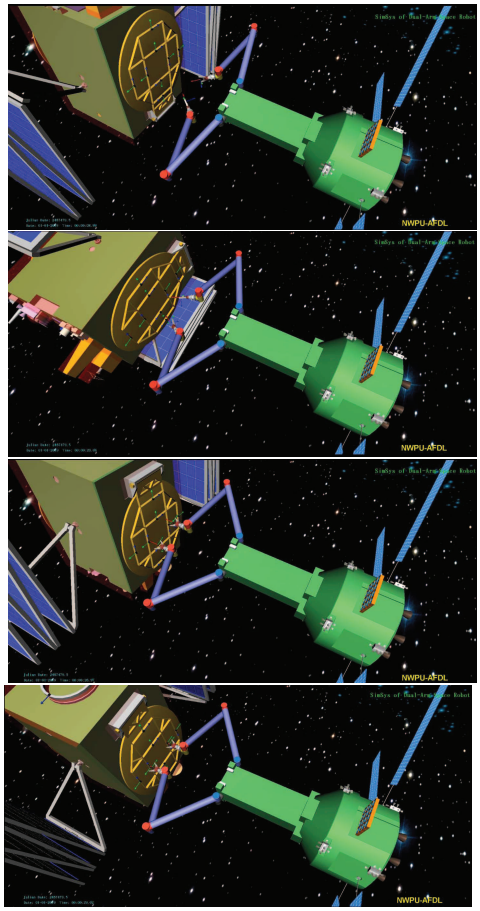

Fig. 9. Simulation results in VR platform

coordinated strategy is needed with the increasing number of agents.

REFERENCES

Abad, A.F. (2014). Optimal control of space robots for capturing a tumbling object with uncertainties. *Journal of Guidance, Control, and Dynamics*, 37, 2014–2017.

Aghili, F. (2013). Pre- and post-grasping robot motion planning to capture and stabilize a tumbling/drifting free-floater with uncertain dynamics. *IEEE International Conference on Robotics and Automation*, 5461–5468.

Ding, W. (2019). An efficient b-spline-based kinodynamic replanning framework for quadrotors. *IEEE Transactions on Robotics*, 35(6).

James, F. (2016). Reactionless maneuvering of a space robot in precapture phase. *Journal of Guidance, Control, and Dynamics*, 39, 2419–2425.

Karaman, S. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30, 846–894.

Kolter, J.Z. (2009). Task-space trajectories via cubic spline optimization. *IEEE International Conference on Robotics and Automation*.

Lampariello, R. (2014). Generating feasible trajectories for autonomous on-orbit grasping of spinning debris in a useful time. *International Conference on Intelligent Robots and Systems*, 5652–5659.

LaValle, S.M. (2006). *Planning algorithms*. Cambridge University Press, London.

Luo, J.J. (2018). A fast trajectory planning framework with task-priority for space robot. *Acta Astronautica*, 152, 823–835.

Nenchev, D. (1992). Analysis of a redundant free-flying spacecraft manipulator system. *IEEE Transactions on Robotics and Automation*, 8, 1–6.

Persson, S.M. (2014). Sampling-based a* algorithm for robot path-planning. *International Journal of Robotics Research*, 33, 1683–1708.

Rybus, T. (2014). Optimal detumbling of defunct spacecraft using space robots. *International Conference on Methods and Models in Automation and Robotics*, 64–69.

Salaris, P. (2019). Online optimal perception-aware trajectory generation. *IEEE Transactions on Robotics*, 35(6).

Vahrenkamp, N. (2010). Integrated grasp and motion planning. *IEEE International Conference on Robotics and Automation*.

Vaz, A. (2006). Tools for robotic trajectory planning using cubic splines and semi-infinite programming. *Lecture Notes in Economics and Mathematical Systems*, 563.

Wang, M.M. (2018). Coordinatd trajectory planning of dual-arm space robot using constrained particle swarm optimization. *Acta Astronautica*, 146, 259–272.

Webb, D.J. (2013). Kinodynamic rrt*: asymptotically optimal motion planning for robots with linear dynamics. *IEEE International Conference on Robotics and Automation*.

Xu, W.F. (2009). Study on non-holonomic cartesian path planning of a free-floating space robotic system. *Advanced Robotics*, 23, 113–143.

Zhu, Z. (2015). A convex optimization approach to smooth trajectories for motion planning with car-like robots. *54th IEEE Conference on Decision and Control*, 835–842.