2022 7th International Conference on Automation, Control and Robotics Engineering (CACRE) | 978-1-6654-0387-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/CACRE54574.2022.9834167

Research on Motion Trajector Planning of Industrial Robot Based on ROS

1st Tian Xu
School of Automation, Nanjing Institute of Technology
Nanjing, China
1005097959@qq.com

2nd Han-Zhong Liu*
School of Automation, Nanjing Institute of Technology
Nanjing, China
zdhxlhz@njit.edu.cn

3rd Rong Huang
School of Automation, Nanjing Institute of Technology
Nanjing, China
2231372322@qq.com

4th Zheng Yao
School of Automation, Nanjing Institute of Technology
Nanjing, China
1044110186@qq.com

5th Ju-Yang Xu
School of Automation, Nanjing Institute of Technology
Nanjing, China
1159935392@qq.com

***Abstract*—In view of the current traditional robot control system with narrow application surface, low code reuse rate, poor scalability, complex model construction, trajectory planning algorithm implementation difficulties, single function and transplantation difficulties, the purpose of this paper is to use the open source robot operating system ROS and to build the robot control system, and the simulation of robot motion planning and experimental verification.First, the Moveit was performed for the URDF model of the robot.The configuration of the function package.Then the trajectory planning of the robot spatial line and spatial arc was conducted, and the path point information of the spatial trajectory was generated according to the OMPL motion planning library of the ROS system.Finally, in order to better control the actual robot, the path point information is interpolated with quintic splines, and the interpolated data was sent to the servo driver through the EtherCAT bus, so as to control the actual robot movement..Finally, through simulation verification, the results show that the motion trajectory of each joint of the robot is stable, with high accuracy, and the curve obtained by the quintic spline is more smooth and stable, which provides a method for the practical motion planning of the robot.**

Keywords: ROS; industrial robot; motion planning; spline interpolation

I . Introduction

The research in the field of robot motion focuses on trajectory planning and precision control. In 2005, the Indian Institute of Technology studied the optimal working speed and the trajectory of robot parts machining by analyzing the grinding effect and the cutting depth of robot machining parts.In 2007, Nagata of Japan studied a motion accuracy of controlling the robot surface motion through the robot joint position and terminal joint force, formed a closed-loop loop through force feedback and force detection device, and continuously corrected the robot joint motion trajectory according to the current joint force. [1]Professor Guo Xiaotong proposed a research method to improve the control of the robot joint position by constantly changing the robot joint position through PD feedback data, and carried out experiments on existing robots.Harbin Institute of Technology[2] solves the robot reverse kinematic optimal solution to find the size of each joint angle under the Cartesian coordinate system, and performs numerical simulation to verify the results.In 2017,Ross A Knepper performed collision detection in trajectory planning by a stochastic sampling method, which not only avoided obstacles in a narrow space, but also performed well in real-time.

Through the analysis of the above research results, most of the robot trajectory planning method is to implement the robot position control in cartesian space. And we can sum up that robot singularity and robot joint limit increase difficulty to the solution of robot reverse kinematics.

With the gradual transformation of China's manufacturing industry from traditional manufacturing mode to intelligent manufacturing mode, the application field of industrial robot is also expanding, and the requirements for robot control technology are becoming higher and higher. The following figure introduces the structure of industrial robot control system.

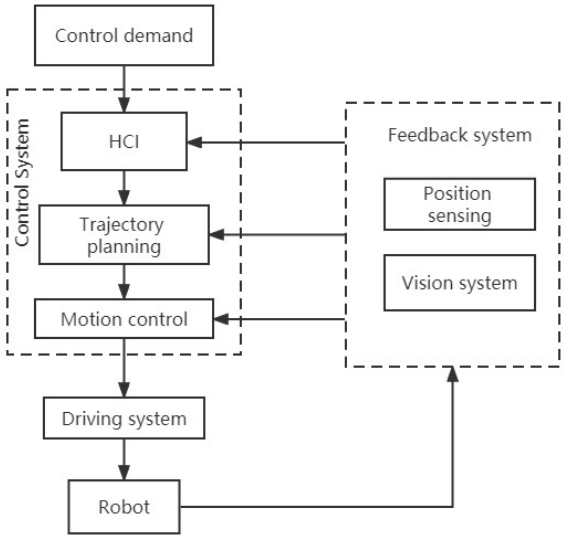


Fig. 1 The structure of industrial robot control system

For the convenience of robot trajectory planning, in 2007 Stanford university and Willow Garage robot company, project development and put forward a series of concepts, after specific research and experiment, in 2010, the MoveIT! function package to solve the robot motion planning library, in OMPL motion planning library contains many spatial algorithm and control algorithm planner, can be applied in the configuration of move IT robot model[3].OMPL can provide different computing methods[4] according to different KDL computing algorithms, which can adapt to the motion calculation of complex algorithms and calculate values with high accuracy, saving a lot of computing time[5] for robot trajectory planning.

Many scholars have begun to perform research on robot motion planning using ROS systems in recent years. Li Hao [6] et al. employed an optimized S-shaped trajectory planning method based on the ROS system, which ensured the smoothness and stability of the robot motion by smoothing variations in joint velocity, acceleration, and acceleration. To simulate and realize collision-free path planning of the robot in difficult settings, Jiang Guo [7] et al. integrated a fast search random tree method with the Motion Planning plug-in of the ROS system. After trajectory design, Rujia Liu[8]et al. employed a quintic polynomial interpolation approach to match the path points and proved the method's performance through simulation testing.

We investigate robot motion planning using the ROS system, using coarse interpolation of linear and circular arc trajectories in the spatial coordinate system and fine interpolation of joint angles in a joint coordinate system using the quintic spline, and we verify the control effect of robot motion planning through simulation and experiment.

II. The construction of the robot operating system

MoveIt! is an easy-to-use integrated development platform in the ROS community, consisting of mobile operational function packages such as motion planning, operational control, 3D perception, kinematics, control and navigation algorithms.In the function package, the move_group node is the control core. Beside other interfaces, the move_group node can call three interfaces, including C++ interface, Python interface and GUI visualization interface.The move_group node also obtains the information of URDF and SRDF.In the process of controlling the robot, the move_group node will interact with the robot controller, and the robot sensor will constantly feedback the joint state information to the move_group node, so that the move_group node can call the motion planning algorithm to solve the positive and inverse kinematic solution and generate the motion trajectory.

The ROS robot operating system is a Linux-based application framework that allows users to control a virtual robot using the MoveIt! Assistant tool interface.The previously developed URDF model of the robot is imported into the Setup Assistant, and the self-collision detection matrix is adjusted such that the robot's linkages do not overlap each other during motion. After that, the robot's base position is set to a virtual joint, and a planning group is constructed to decide which connections and joints need to be planned. The robot's zero position is finally set, and the configuration file generation is finished.Fig.2 depicts the Moveit! function package's configuration flow.

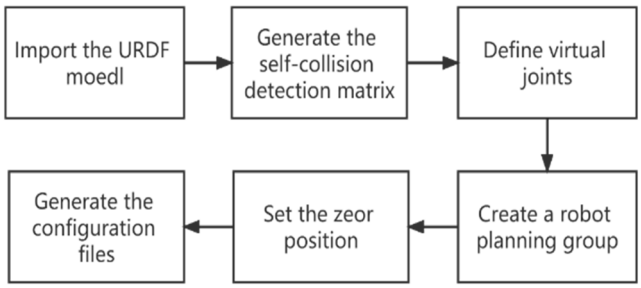


Fig. 2 Configuration process of the Moveit !function package

III. Track planning of the robot

After the ROS robot operating system is built, coarse interpolation of linear orcircular trajectories in the spatial coordinate system is performed, and the coarse interpolation points are written into the OMPL motion planning library, maximum velocity and acceleration are set, angular information and corresponding time information of the robot's six joints are generated, and quintic polynomial interpolation in the joint coordinate system is performed. To control the virtual robot motion, the fine interpolated path point information is given to the joint trajectory controller, which is published to the virtual joints in the Rviz interface via joint mapping and topic publishing.The overall control flow chart is shown in the figure below.

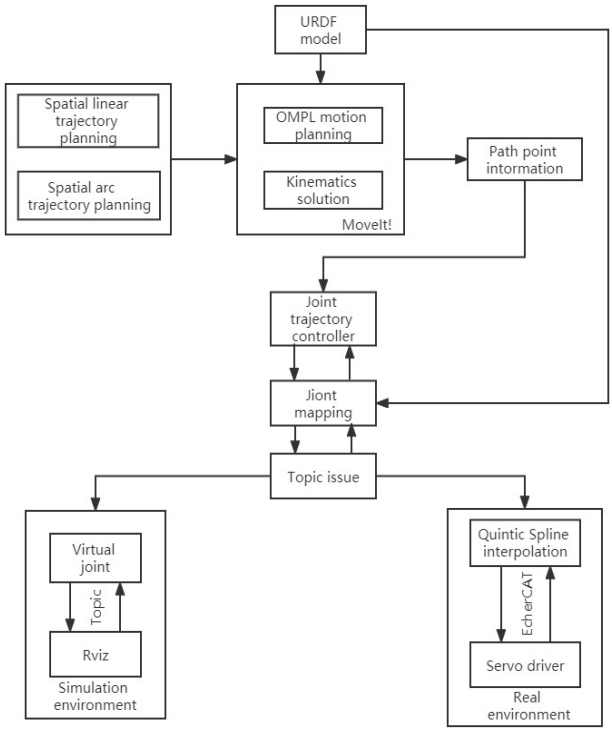


Fig. 3 Overall control flow chart

A. Coarse interpolation of spatial linear trajectories

In Cartesian space, when the robot's end actuator moves along a straight line from $P_0(x_0, y_0, z_0)$ to $P_e(x_e, y_e, z_e)$, the distance between them needs to be calculated with a value of L:

$$\boldsymbol{L = \sqrt{(x_e - x_0)^2 + (y_e - y_0)^2 + (z_e - z_0)^2}} \quad (1)$$

Assuming that the robot end actuator moves from point P_0 to point P_e at speed v, where the interpolation time interval Δt is taken, so the distance of the adjacent interpolation points is : $\Delta L = v * \Delta t$ .Thus, the insertion number of the spatial line trajectory planning can be calculated as:

$$\boldsymbol{N = Ent\left(\frac{L}{\Delta L+1}\right)} \quad (2)$$

In Equation (2), Entmeans the integer function is taken down.

It can be concluded that the coordinates $(x_{i+1}, y_{i+1}, z_{i+1})$ of the intermediate interpolation points of the spatial line trajectory are:

$$\begin{cases} \boldsymbol{x_{i+1} = x_0 + \frac{x_e - x_0}{N} i} \\ \boldsymbol{y_{i+1} = y_0 + \frac{y_e - y_0}{N} i} \\ \boldsymbol{z_{i+1} = z_0 + \frac{z_e - z_0}{N} i} \end{cases} \boldsymbol{, i = 0, 1, 2, \dots, N} \quad (3)$$

B. Coarse interpolation of spatial circular arc trajectory

In Cartesian space, when the robot end actuator makes spatial arc trajectory planning, the three noncol-linear points must be determined in space,whose coordinates are $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$,and $P_3(x_3, y_3, z_3)$ .This three points can determine a plane, and then make the vertical bisector of the line P_1P_2 and line P_2P_3, the point $O_1$is the center coordinate of the arc, whose coordinates are$O_1(x_4, y_4, z_4)$.Thus, the radius R of the arc is:

$$\boldsymbol{R = \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2 + (z_4 - z_1)^2}} \quad (4)$$

Known center coordinates and spatial three-point coordinates, easy to find the corresponding arc P_1P_2 and arc P_2P_3 ,center angle is θ_1 and θ_2 .Then, the coordinate dot $O_1$is used to establish a new coordinate system $O_1x_1y_1z_1$, in which the unit vector corresponding to the axis x_1, axis y_1 and axis $z_1$is $\boldsymbol{n, o, a}$.The transformation matrix from the coordinate system $O_0$to the coordinate system $O_1$is:

$$\boldsymbol{T = \begin{bmatrix} n & o & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} nx & ox & ax & px \\ ny & oy & ay & py \\ nz & oz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}} \quad (5)$$

In Equation (5), $\boldsymbol{p = [x_4\ y_4\ z_4]^T}$ .If the angular displacement of the end actuator of the robot is set as $\Delta\theta$, the number of spatial arc interpolation is:

$$\boldsymbol{N = Ent(\frac{\theta_1 + \theta_2}{\Delta\theta + 1})} \quad (6)$$

In Equation (6), Ent means the integer function is taken down.The interpolation of the spatial arc in the plane coordinate system x_1z_1 is:

$$\begin{cases} \boldsymbol{x_{i+1} = x_i \cos\Delta\theta - z_i \sin\Delta\theta} \\ \boldsymbol{z_{i+1} = z_i \cos\Delta\theta + x_i \sin\Delta\theta} \end{cases} \quad (7)$$

In Equation (7), $i = 1,2,3, \dots, N$.For any point in the plane x_1z_1, it can be converted under the base coordinate system $O_0x_0y_0z_0$by Equation (5).With this change matrix, the arc trajectory under the new coordinate system can be calculated, and then the spatial arc trajectory in the Cartesian space can be obtained.

C. OMPL motion planning

After the spatial line or spatial arc trajectory planning, the OMPL (Open Motion Planning Library) motion planning library is used to pick up the generated spatial line or spatial arc.Then based on the RRT(Rapidly-exploring Random Trees) algorithms and positive and inverse solvers of KDL.We can calculate the corresponding spatial line or spatial arc trajectory.However, this spatial trajectory contains only positional information and does not contain velocity and acceleration information.To this end, this trajectory needs to be brought into the Add Time Parameterization module, which functions to set speed and acceleration constraints for this spatial trajectory, and to calculate the time information of the robot moving to each path point to obtain the path point information of the robot.

D. Quintic polynomial interpolation of joint coordinates

To subdivide the angle values of each joint of the robot, the joint coordinate system uses a quintic polynomial interpolation algorithm, and after interpolation, it not only makes the actual generated trajectory trace smoother and keeps the continuity of position and velocity, but it also makes the acceleration not to produce abrupt changes. The following describes the quintic polynomial interpolation algorithm, whose expression is as:

$$\boldsymbol{\theta(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3 + \alpha_4 t^4 + \alpha_5 t^5} \quad (8)$$

Its velocity and acceleration expressions are:

$$\boldsymbol{\dot{\theta}(t) = \alpha_1 + 2\alpha_2 t + 3\alpha_3 t^2 + 4\alpha_4 t^3 + 5\alpha_5 t^4} \quad (9)$$

$$\boldsymbol{\ddot{\theta}(t) = 2\alpha_2 + 6\alpha_3 t + 12\alpha_4 t^2 + 20\alpha_5 t^3} \quad (10)$$

For trajectory planning with high trajectory accuracy requirements, the acceleration constraint must be added, and the rotation angle of a single joint starting point is θ_0, The end-point rotation angle is θ_f, The speed of start and end are expressed by $\dot{\theta}_0$ and $\dot{\theta}_f$, and the acceleration of start $\ddot{\theta}_0$ and end $\ddot{\theta}_f$. $t_f$represents the time to reach the end.The constraints for a point-to-point (PTP) trajectory composed of only two points are: $\theta(0)=\theta_0, \theta(t_f)=\theta_f, \dot{\theta}_0=0, \dot{\theta}_f=0, \ddot{\theta}_0=0, \ddot{\theta}_f=0$.

Therefore, the specific expression of quintic polynomial interpolation which can be obtained by Equation (8), Equation (9), Equation (10) and constraints is :

$$\boldsymbol{\theta(t) = \theta_0 + \frac{10(\theta_f - \theta_0)}{tf^3} t^3 + \frac{15(\theta_0 - \theta_f)}{tf^3} t^4 + \frac{6(\theta_f - \theta_0)}{tf^5} t^5} \quad (11)$$

The interpolation algorithm can be used to calculate the quintic polynomial interpolation joint trajectories through any two points using the above derivation; however, the higher the

number of polynomials, the more accurate the trajectory obtained by the fitting method; however, as the number of polynomials increases, the number of constraints increases, and the calculation becomes more complicated; therefore, this paper uses the quintic polynomial interpolation algorithm for trajectory planning.

IV. Simulation and experiment

On the ROS system, to verify the utilization of Moveit!Function package to control the virtual robot to do space straight line and space arc motion function, using Moveit!The Python related API is provided to write the relevant control program, and finally shows the spatial line and spatial arc trajectory of the virtual robot motion on the Rviz interface, as shown in Fig. 4. and Fig. 5.

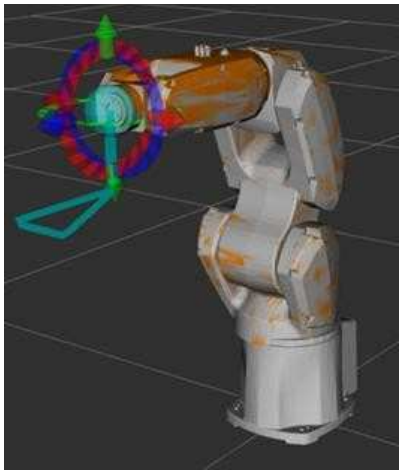


Fig. 4 Simulation of the spatial straight lines

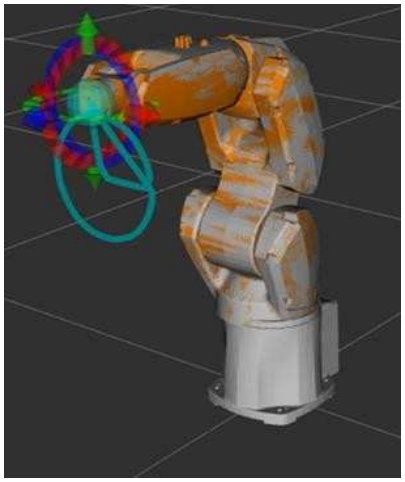


Fig. 5 Simulation of the spatial arcs of virtual robots

V. Physical experiment verification

In order to verify the function of using ROS system to control the motion planning, an industrial control machine and a robot with six degrees of freedom.The specific experimental platform is shown in Fig. 6.

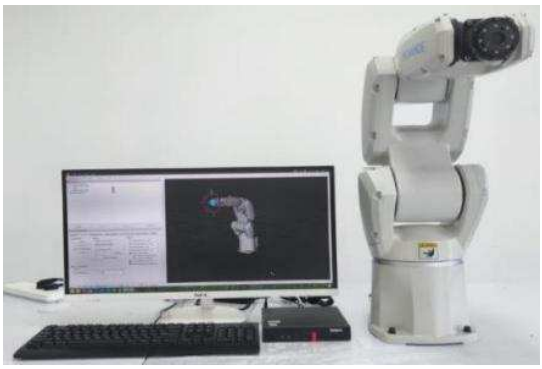


Fig. 6 Experimental Platform

Based on this experimental platform, the overall block diagram of the robot control system presented in this paper is shown in Fig.7.First of all,in terms of industrial control machines, the Industrial control machine is installed into the Linux system.Then,in order to improve the real-time performance of the system, configurate it with Motorcortex live,then make it as a IGH EtherCAT master station in order to make the industrial control machine directly communicate with the industrial robot through the network port by Ethercat.Last,we need install into ROS operating system, and transform the IGH EtherCAT main station into a node module under the ROS system, in order to send the state parameters of the robot to the IGH EtherCAT main station through the hardware interface of the ROS system, and realize the process data distribution and receiving in the real-time core of the main station.In terms of industrial robots, this paper selects industrial robots with six degrees of freedom, which mainly includes six servo drivers and the corresponding servo motors.

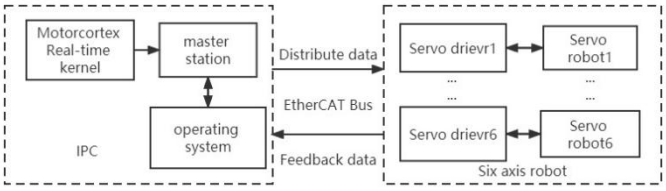
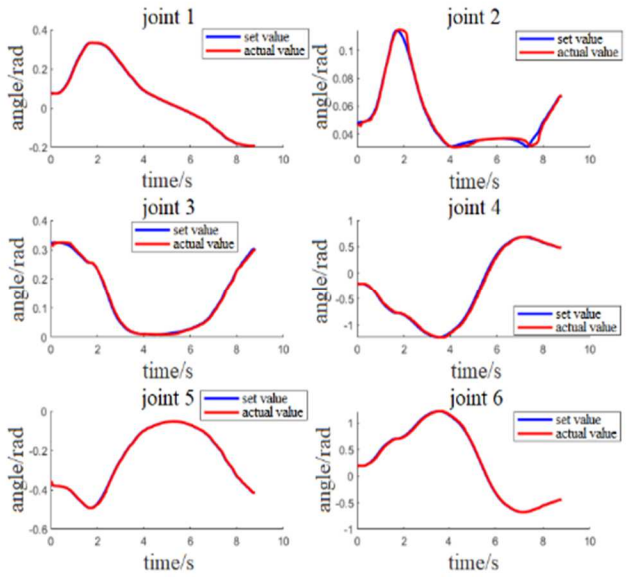
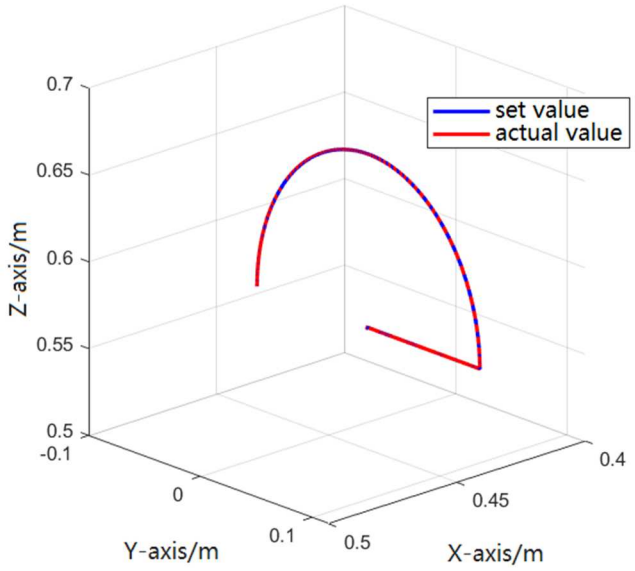


Fig. 7 Overall block diagram of the robot control system

Taking the robot space arc trajectory planning as an example, the pair of set value of the robot and the actual value of each joint angle is shown in Fig. 8 (a), and the pair of set value of the spatial trajectory at the end of the robot is shown in Fig. 8 (b).As can be seen from Fig. 8, in the actual process of robot movement, the movement of the robot is smooth, and the distance error between the tracking space arc and the theoretical space arc is also small, which ensures the high accuracy of the robot trajectory and verifies the effect of robot motion planning based on ROS system.



(a) Angle of the robot



(b) Robot end spatial trajectory

Fig. 8 Comparison between the setting value and actual value of robot spatial arc trajectory planning

VI. Conclusion

In this study, we used the ROS system to simulate and experiment with robot motion planning. First of all, we generated the URDF model of the robot and completed the parameter configuration of Moveit!. The path point information is then generated by planning the robot's spatial straight line and spatial circular arc trajectory, using the OMPL motion planning library to take the points of the planned spatial trajectory, and assigning velocity and acceleration restrictions to this spatial trajectory. Finally, the path point information is processed by quintic polynomial interpolation technique to distribute the planned path point information to the solid robot, and the testing results demonstrate that the robot moves smoothly and accurately under the joint coordinate system. Simultaneously, the way of operating the robot in this research based on the ROS system causes the virtual robot and the actual robot to move simultaneously, providing an important reference for evaluating the robot's digital twin function.

References.

[1] Nagata F, Hase T, Haga Z, et al. CAD/CAM-based position/force controller for a mold grinding robot [J].Mechatronics, 2007.17(4-5):207-216.

[2] She Y. Modeling and fault-tolerant control of a redundant robot arm under single joint failure[D].Harbin Institute of Technology, 2013.

[3] Hao Qi, Xu Shixiang, Jiang Tianqi, Fang Jianzhong. ROS-based robotic arm combined with RGBD camera obstacle avoidance motion planning [J].Automation in Manufacturing Industry, 2019,41 (04): 56-60.

[4] Chengren Yuan,Guifeng Liu,Wenqun Zhang,Xinglong Pan.An efficient RRT cache method in dynamic environments for path planning[J].Robotics and Autonomous Systems,2020,131.

[5] Liu Lei, Ning Yi.Six-DOF robotic arm trajectory planning based on ROS [J].Automation and instrumentation, 2018,33 (03): 22-25 + 51.

[6] Li Hao, Zheng Zhizhen,Huang Shunzhou,Yang Fuwei,Liu Zeyu. ROS-based trajectory planning and simulation for industrial robots[J]. Combined Machine Tools and Automatic Machining Technology,2018(12):59-62.

[7] Guo JG,ShuangFeng,Kan HL. Research on ROS-based motion planning of six-joint robot[J]. Journal of the Anhui University of Engineering, 2019,34(01):36-45.

[8] Liu Rujia,Wang Fang,Zhang Qiang,Li Nan. Research on ROS-based robotic arm trajectory planning[J]. Navigation Positioning and Timing, 2016, 3(06):82-88.