A Study of Robot Arm Control System Based on ROS

Mu Weiwei

Zhonghuan Information College Tianjin University of Technology, Tianjin, 300380, China

Keywords: ROS; Robot Arm; Motion Planning

Abstract: Aiming at the problem of large workload and poor portability of traditional robotic control system when facing specific objects, an internationally popular open source robotic operating system (ROS) was used as a platform to construct a robotic control system, and a self-made SCARA manipulator was used as an example to illustrate the system design method. URDF format files were used to import robot data and display the three-dimensional model on the ROS visualization tool RViz. The MoveIt! was used to complete the motion planning of the manipulator, and to speed up the operation through a custom inverse algorithm. Finally, according to the specific hardware, the PVT motion data was converted into actual motion. The experimental results show that the robot control system based on ROS can be designed quickly and efficiently to meet working requirements.

1. Introduction

Robot technology is a new and high technology which integrates many disciplines and its development level is often regarded as an important symbol of a country's industrial level. With the rapid development of electronic technology, computer and software in recent years, robots have also made remarkable progress in automation and intellectualization, and their role in modern society has become more and more important. However, the complexity of robotic systems and the diversity of tasks have brought many difficulties to the programming of robots. In the past, based on independent system platforms and software architectures, transplanting the same algorithm between different robots also takes a lot of labor. As the most important actuator of robots, robots are endowed with the ability of physical interaction with the external world environment. This makes the intelligence of the robot not only stay on the recognition and planning level, but also express it through the actual operation. At present, the research hot spots of robotic arm at home and abroad mainly focus on the autonomous and precise grasping and placement of objects. However, it is very difficult for the manipulator to perform a series of complex and precise operations perfectly so that the end effector can reach the predetermined position, which requires a lot of mathematical knowledge related to the manipulator solution.

However, the development of robotic arms based on ROS and MoveIT! will be relatively simple for developers. ROS (Robot Operating System) makes code reuse and modular design simpler by providing a unified software platform, which can significantly improve the speed of robot application development. MoveIt! is state of the art software for mobile manipulation, incorporating the latest advances in motion planning, manipulation, 3D perception, kinematics, control and navigation. It provides an easy-to-use platform for developing advanced robotics applications, evaluating new robot designs and building integrated robotics products for industrial, commercial, R&D and other domains. In this paper, ROS-based control of SCARA manipulator was used to achieve such tasks as trajectory planning.

2. System Hardware Architecture

The mechanical platform is shown in figure 1 and will be used for tomato fruit harvesting in the later stage of the project [1]. Two SCARA robot arms are fixed on the same rotating platform. Each robot arm consists of a vertical lifting joint and two rotating joints in series. The initial orientation angle of the two robot arms is 90 degrees. In the overlapping area of the two arms, the cooperative action of the two arms can be fully utilized to complete complex tasks [2]. The end of the manipulator is equipped with flange surface, which facilitates the installation of different types of end effector.

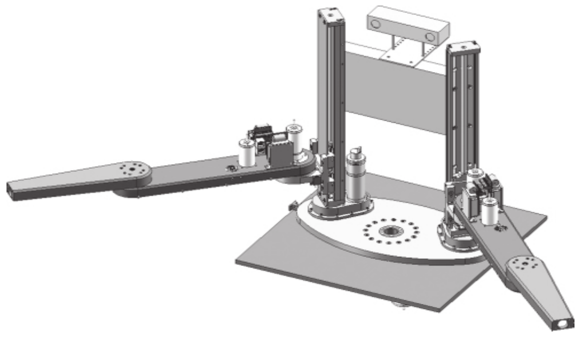


Fig.1. Robot overall structure diagram

The whole mechanical system is driven by seven DC servo motors, and the Gold series motor drivers and controllers of ELMO Company in Israel are selected [3]. The drives are connected by a chrysanthemum chain and connected to the controller. The EtherCAT bus protocol is used to communicate within the drives, which has a high data synchronization rate [4]. Motion controller adopts Linux system with hard real-time expansion, and can realize multi-axis linkage control with driver. Motion controller communicates with main industrial computer through Modbus TCP protocol. The overall structure of the control system is shown in Figure 2.

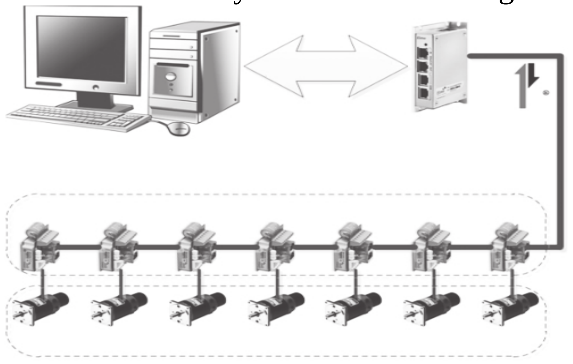


Fig.2. Overall structure diagram of control system

3. Application and Implementation

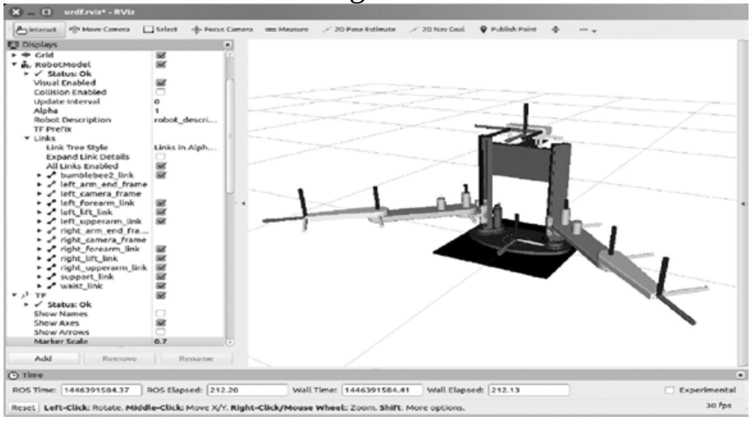
3.1. Introduction to ROS

ROS is a secondary operating system for robots, which essentially provides a distributed processing framework and needs to run on the main operating system (usually Ubuntu). Under the framework of ROS, different functional modules can be designed separately and run as ROS nodes. Multiple ROS nodes can realize point-to-point communication based on TCP/IP sockets, mainly including heterogeneous data flow communication based on topic message and synchronous RPC (remote procedure call) communication based on service [5]. Different nodes can run on multiple computers connected through the network. ROS communication is language-independent. Node programs support multi-language implementation, including C++, Python and so on. Node Manager (master) is responsible for storing the subject and service information of all nodes. It establishes a point-to-point direct connection query Table between two nodes, which acts like DNS domain name server.

3.2. Robot model leading-in

In ROS, the unified robot description format (URDF) is used to Abstractly describe the hardware model of the robot with tree structure. URDF is written based on XML, in which the robot is Abstracted as a connecting rod structure connected by various joints. For each link, its appearance attributes, including shape and color, can be displayed in the visualization tool RViz. The shape of the link can be described by simple statements, expressed by cuboids or cylinders, or pointed to external grid data (STL or Collada format) to accurately represent. For each joint, it is necessary to point out the two connecting rods, and include the position and orientation information of the joint axis in the parent linkage coordinate system [6].

Firstly, the three-dimensional model of the robot constructed in SolidWorks is derived into STL format, and the parameters of the nominal link of the robot are measured and written into the URDF file. Based on the attributes of the robot model in URDF and the current joint values, ROS will automatically generate the transformation relationship between the two connecting rod coordinate systems connected by each joint, and then broadcast these data through the TF node [7]. Robot links are connected by tree structure, so the transformation relationship between any two links coordinate systems can be calculated by these data. The display effect of the robot model and the connecting rod coordinate system under RViz is shown in figure 3.



Grid	✓
RobotModel	✓
Status: Ok	
Visual Enabled	✓
Collision Enabled	☐
Update Interval	0
Alpha	1
Robot Description	robot_descri...
TF Prefix	
Links	
Link Tree Style	Links in Alph...
Expand Link Details	☐
All Links Enabled	✓
bumblebee2_link	✓
left_arm_end_frame	
left_camera_frame	
left_forearm_link	✓
left_lift_link	✓
left_upperarm_link	✓
right_arm_end_fra...	
right_camera_frame	
right_forearm_link	✓
right_lift_link	✓
right_upperarm_link	✓
support_link	✓
waist_link	✓
TF	✓
Status: Ok	
Show Names	☐
Show Axes	✓
Show Arrows	☐
Marker Scale	0.7

ROS Time: 1446391584.37 ROS Elapsed: 212.20 Wall Time: 1446391584.41 Wall Elapsed: 212.13 ☐ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel: Zoom. Shift: More options. 30 fps

Fig.3. Robot model and coordinate system displayed in RViz

3.3. Motion planning

ROS provides a powerful integrated function package for manipulator control, namely MoveIt! [8]. It involves motion planning, robot forward and inverse kinematics, collision detection, robot arm grasping and other fields [8]. The robot to be controlled in this paper is a two-arm structure. It needs to design a controller for each manipulator. Taking the left arm as an example, the coordinate system in figure 4 is given. The height Z has a simple linear relationship with joint angle, so it is not displayed in the formula.

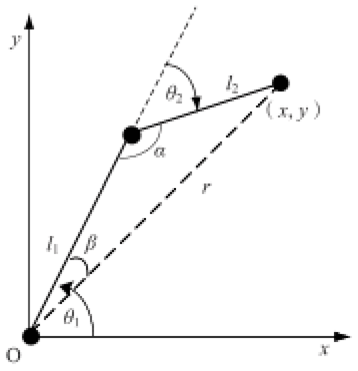


Fig.4 The coordinates of the end of the left arm

The trajectory planning of the manipulator needs to be carried out in joint space or Cartesian space according to the specific task requirements [9]. Generally, only the starting point and the end point are given, and the intermediate key point or other path constraints need to be added in specific cases. ROS uses OMPL, a random sampling based motion planning algorithm library, to complete trajectory planning. The planned trajectory avoids collision states such as obstacles and self-interference, as shown in figure 5.

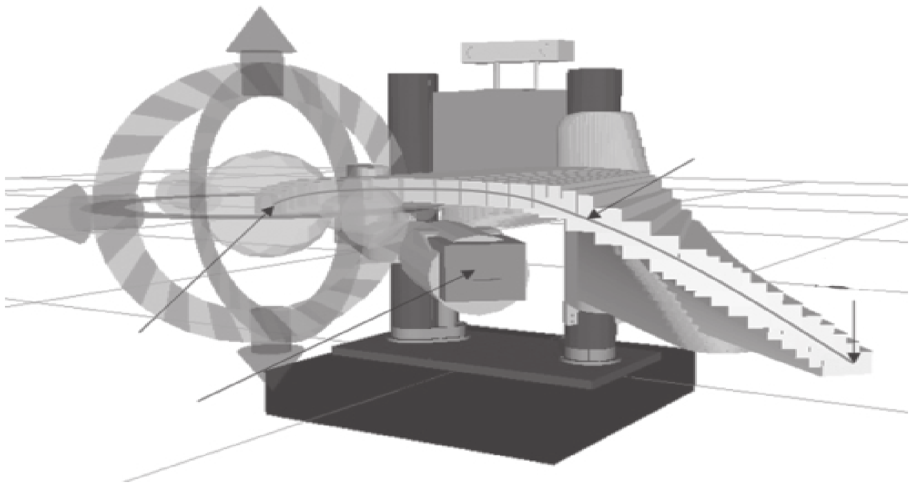


Fig.5. Trajectory planning of robot arm

If the trajectory planning is carried out in Cartesian space, the inverse kinematics of the robot needs to be transformed into the representation in joint space. MoveIt! uses KDL library to solve inverse kinematics problem by numerical iteration algorithm. This method has strong universality and adaptability to different robot models, but the iteration method is slow in calculation. So MoveIt! also provides interface to enable users to adopt self-defined inverse kinematics algorithm. Similarly, taking the left arm as an example, the inverse kinematics analytical solution formula of the manipulator can be obtained by a geometric method.

3.4. Motion control

The motion data of the manipulator obtained after motion planning are given in PVT format. PVT refers to position, velocity, and time. They correspond to the position, instantaneous velocity and time needed to reach the position of each joint of the manipulator. Its mathematical essence is piecewise cubic Hermite interpolation [10]. As shown in figure 2, the industrial computer transmits the PVT motion data to the multi-axis controller through the Modbus TCP protocol. The multi-axis controller completes the interpolation operation, and the Ether-CAT real-time bus controls each axis to complete the predetermined action.

4. Operation Test

After the design of control system based on ROS on the experimental platform of the robot arm, several operation tests have been carried out. ROS provides a convenient interactive interface. The towed ball shown in figure 5 can be used to intuitively select the terminal position and provide off-line motion simulation. It is helpful for the initial functional verification experiment to perform the actual action after confirmation. Programming control method is mostly used to realize on-line control in practical application. For this ROS, C and Python interfaces are provided. Programming control method can also achieve more complex motion requirements by adding intermediate key points and adding path constraints. The test results show that the control system can quickly plan the appropriate trajectory and control the manipulator to achieve action for the reasonable task requirements, which fully meets the actual work requirements.

5. Conclusion

With the development of science and technology, robots will be more widely used. This paper introduced a method of designing robot arm control system based on ROS, and an open source robot operating system. The system design can be completed quickly by applying the latest open source software achievements. In addition, ROS is also widely used in robotic navigation, machine vision, robotic cluster control, and so on. Aiming at the problem of large workload and poor portability of traditional robot control system when facing specific objects, this paper constructed a robot control system based on the internationally popular open source robot operating system (ROS), and illustrated that the system can display a three-dimensional model on the ROS visualization tool RViz by taking the self-made SCARA manipulator as an example. The use of MoveIt! as a design method and URDF format file can import the robot data, complete the motion planning of the manipulator, and accelerate the operation speed through the self-defined inverse algorithm. Finally, according to the specific hardware, PVT motion data can be converted into actual motion. The experimental results show that the robot arm control system can be designed efficiently based on ROS.

References

[1] Wang Tianmiao, Tao Yong. Present Situation of Industrial Robot Technology and Industrial Development in China. Journal of Mechanical Engineering, 2014(9): 1-13.

[2] Tan Min, Wang Shuo. Research Progress of Robot Technology. Journal of Automation, 2013, 39(7): 963-972.

[3] Quigley M, Conley K, Gerkey B, et al. ROS: An Open-source Robot Operating System. ICRA Workshop on open source software, 2012.

[4] Jansen D, Buttnerh. Real-time Ethernet: The Ether CAT solution. Computing and Control Engineering, 2014, 15 (1): 16-21.

[5] Zhang Jianwei, Zhang Liwei, Hu Ying, et al. Open Source Robot Operating System - ROS. Beijing: Science Press, 2012.

[6] Cao Zhengwan, Ping Xueliang, Chen Shenglong, etc. Robot model building method based on ROS. Modular machine tools and automatic processing technology, 2015(8): 51-54.

[7] FOOTE T. The Transform Library. Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference on. IEEE, 2013: 1-6.

[8] Chitta S, Sucan I, Cousins. MoveIt! [ROS topics]. IEEE Robotics &amp; Automation Magazine, 2014, 1(19): 18-19.

[9] Sucan I, Moll M, Kavraki L E. The Open Motion Planning Library. IEEE Robotics &amp; Automation Magazine, 2015, 19(4): 72-82.

[10] Mao Ziming, Zhang Weijun, Yuan Jianjun, et al. Spline interpolation based on PVT model. Mechatronics, 2016(5): 25-27