

Jaka_ros 使用说明

版本：V2.2

时间：2023.8.8

文档信息

应用领域			
文件编号		发布日期	
起草人		起草日期	2021.10.26
复审人		复审日期	
批准人		批准日期	

版本记录

版本编号	版本日期	修改者	说明
v2.0	2022.7.1		SDK 版本: V2.1.2.4_dev
v2.1	2023.5.17		SDK 版本: V2.1.2_stable_linux
v2.2	2023.8.8		SDK 版本: V2.1.4_8dev-Ming-7_linux

产权说明

节卡机器人股份有限公司 版权所有。

节卡机器人股份有限公司对本文档中介绍的产品所包含的相关技术拥有知识产权。

本文档及相关产品按照限制其使用、复制、分发和反编译的许可证进行分发。未经节卡机器人股份有限公司事先书面授权，不得以任何方式、任何形式复制本产品或本文档的任何部分。

目录

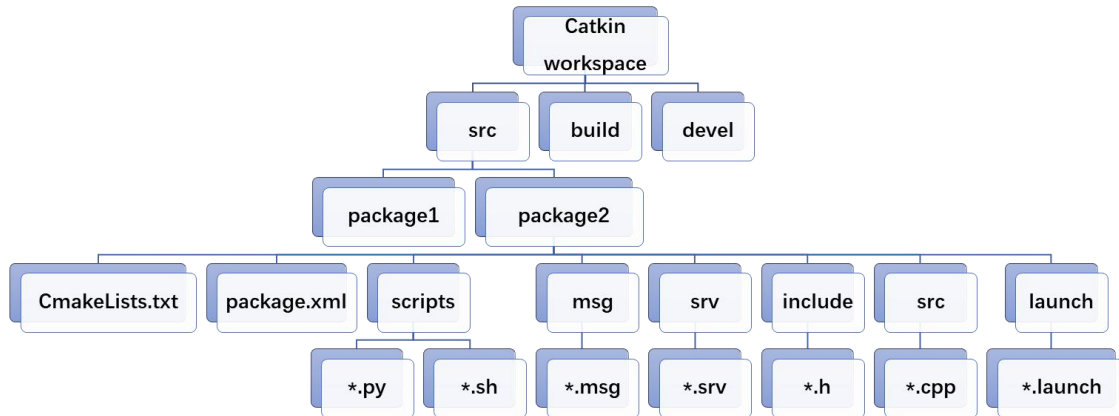
第 1 章 ROS 简介.....	5
1.1 ROS 功能包简介.....	5
1.2 ROS 常用指令.....	6
第 2 章 ROS 控制 JAKA 机器人.....	7
2.1 准备工作.....	7
2.1.1 安装 ROS.....	7
2.1.2 下载 JAKA ROS 功能包.....	8
2.1.3 重要注意事项.....	8
2.2 控制 JAKA 机器人.....	9
2.2.1 设置终端环境.....	9
2.2.2 Jaka_Ros 驱动接口.....	10
2.2.3 Moveit 和 Gazebo 联合使用.....	15
2.2.4 Moveit 和真实机器人联合使用.....	17
第 3 章 JAKA 机器人 ROS 功能包应用案例.....	21

第 1 章 ROS 简介

1.1 ROS 功能包简介

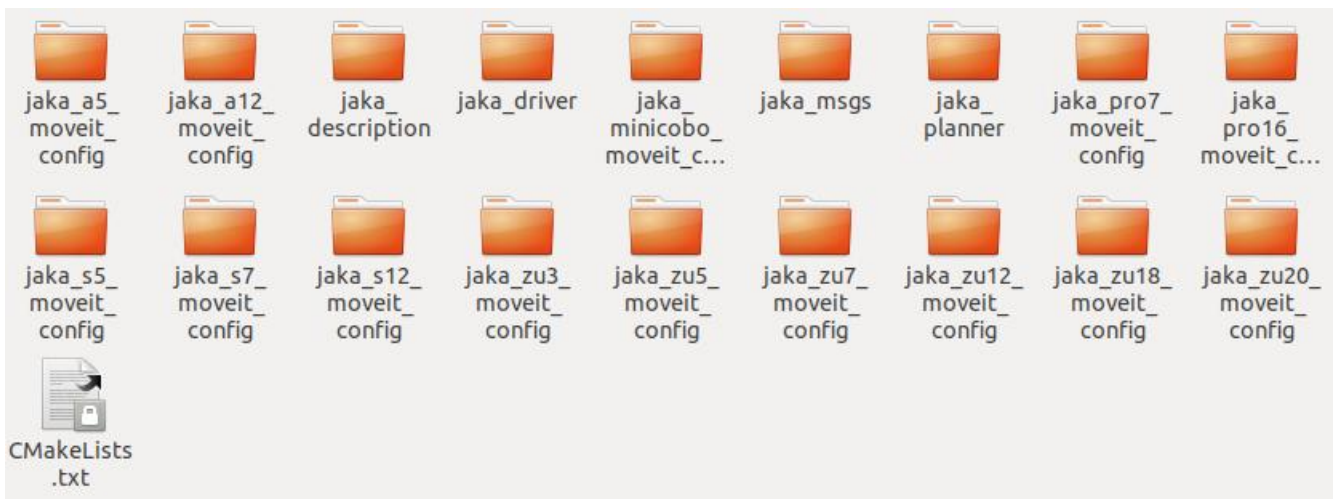
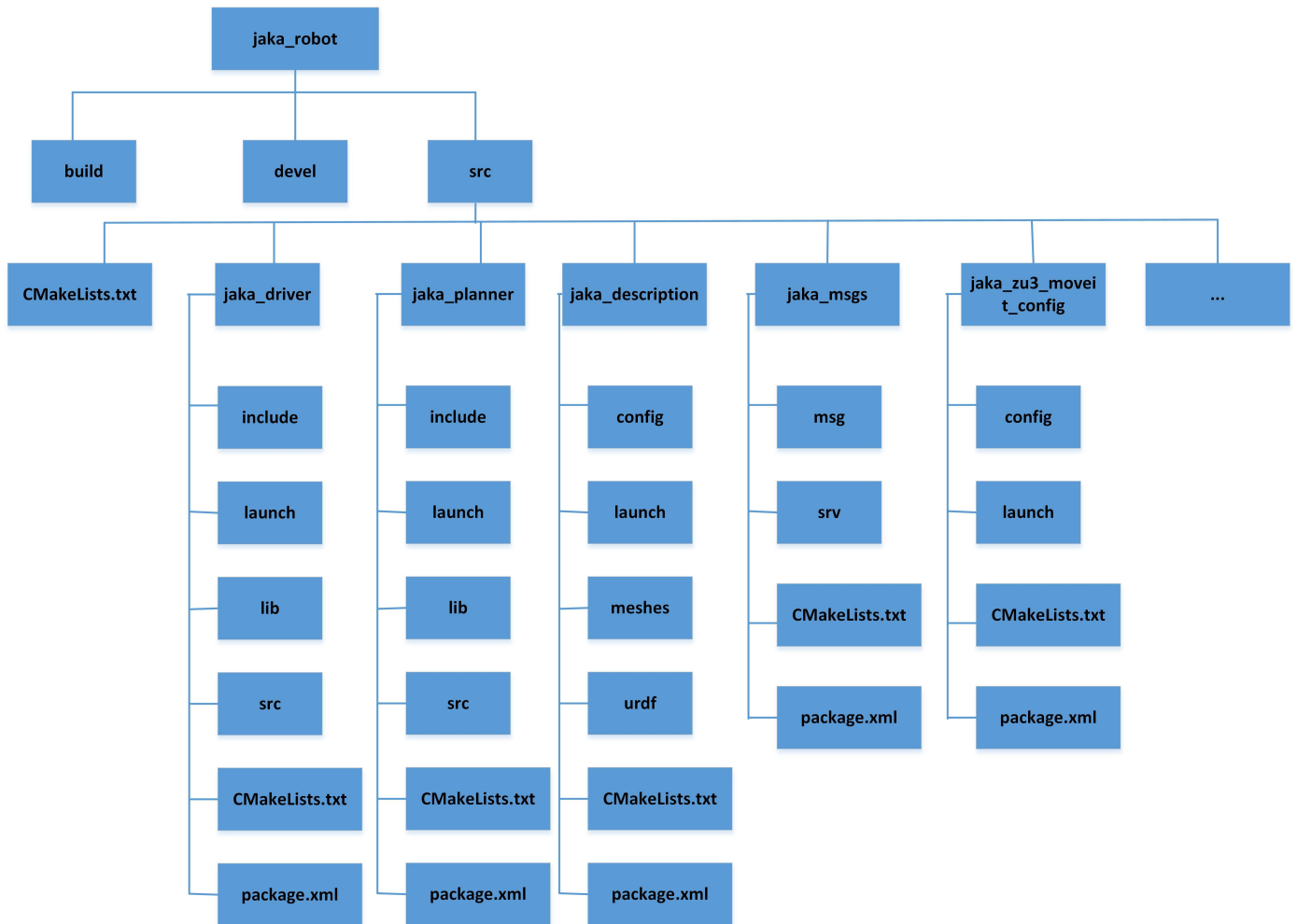
ROS (Robot Operating System) 是适用于机器人的开源元操作系统。ROS 集成了大量的工具、库、协议，提供类似 OS 所提供的功能，可以实现对机器人的控制。

ROS 文件系统指的是在硬盘上 ROS 源代码的组织形式，其结构大致如下图所示：



JAKA 机器人的 ROS 功能包结构大致如下图所示：

注意：工作空间（jaka_robot）中的 build 文件夹和 devel 文件夹，需要重新编译才能生成。



各功能包说明如下:

名称	说明
jaka_decription	机器人模型描述文件（urdf）
jaka_driver	连接真实机器人的启动功能包
jaka_jog_panel	Jog 运动控制面板文件
jaka_msgs	自定义message 类型
jaka_planner	MoveIt 路径规划包
jaka_minicobo_moveit_config jaka_pro7_moveit_config jaka_pro16_moveit_config jaka_zu3_moveit_config jaka_zu5_moveit_config jaka_zu7_moveit_config jaka_zu12_moveit_config jaka_zu18_moveit_config jaka_a5_moveit_config jaka_a12_moveit_config jaka_s5_moveit_config jaka_s7_moveit_config jaka_s12_moveit_config	各型号（minicobo、pro7、pro16、zu3、zu5、zu7、zu12、zu18、a5、a12、s5、s7、s12）机器人 Moveit 配置文件

1.2 ROS 常用指令

常用命令	含义
roscore	启动节点管理器
cd ~/catkin_ws catkin_make	编译 ROS 程序
source ./devel/setup.bash	添加程序包到环境变量中
rospack list	查看软件包列表和定位软件包
rospack find package-name	寻找一个软件包的目录
roslaunch package-name executable-name	启动节点
rostopic list	查看节点列表
rostopic info node-name	查看特定节点信息
rostopic kill node-name	终止节点
rostopic -h	查看 rostopic 所有操作
rostopic list	查看所有 Topic 列表
roslaunch rqt_graph rqt_graph roslaunch rqt_plot rqt_plot	图形化显示 topic
rostopic echo [topic]	查看某个 Topic 信息
rostopic type [topic] rostopic show [msg_type]	查看 Topic 消息格式
rosservice -h	查看所有 service 操作
rosservice list	查看 service 列表
roslaunch package name file.launch	运行 launch 文件

第 2 章 ROS 控制 JAKA 机器人

2.1 准备工作

2.1.1 安装 ROS

以 x86_64 架构 Ubuntu 18.04 为例，在 Ubuntu 18.04 上安装 ROS 的步骤如下：

1. 设置 Ubuntu 软件仓库：

打开终端，运行以下命令以设置添加 ROS 软件仓库到系统中：

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu bionic main" > /etc/apt/sources.list.d/ros-latest.list'
```

2. 添加 ROS 密钥：

继续在终端运行以下命令以添加 ROS 密钥：

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

3. 更新软件包索引：

执行以下命令来更新软件包索引：

```
sudo apt update
```

4. 安装 ROS：

现在可以安装 ROS 了。可以选择不同的安装配置，取决于用户需求。以下是完整安装 ROS 桌面完整版（包括 ROS、rqt、rviz 等）的步骤：

```
sudo apt install ros-melodic-desktop-full
```

请注意，这里使用的是 ROS Melodic 版本，适用于 Ubuntu 18.04。如果你希望安装其他版本的 ROS，只需将命令中的"melodic"替换为其他版本的代号（如"noetic"、"kinetic"等）。

5. 初始化 ROS 环境：

安装完成后，需要初始化 ROS 环境。在终端运行以下命令：

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

6. 安装 rosdep：

rosdep 可以帮助安装 ROS 软件包的依赖项。在终端运行以下命令：


```
sudo apt install python-rosdep
sudo rosdep init
rosdep update
```

7. 创建工作空间：

需要创建一个 ROS 工作空间来组织和构建自己的 ROS 软件包。在终端运行以下命令来创建一个工作空间（假设工作空间名为"catkin_ws"）：

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/
catkin_make
```

8. 开始使用 ROS：

现在已经成功安装了 ROS 并创建了一个工作空间。可以开始编写、构建和运行 ROS 软件包了。

请注意，以上步骤仅为安装和初始化 ROS 的基本步骤。具体的 ROS 开发和使用过程可能涉及到更多的操作和设置。可以参考 ROS 官方文档（<http://wiki.ros.org/>）以获取更多详细信息和指导。

2.1.2 下载 JAKA ROS 功能包

JAKA 机器人的功能包（源代码）下载地址：

```
https://github.com/JakaCobot/jaka\_robot
```

2.1.3 重要注意事项

1. 目前只支持 ROS1 版本。
2. 目前只适用于x86_64架构下。
3. JAKA_ROS 包支持的 ROS 版本为 melodic 和 noetic。手册案例是在 melodic 版本下测试的。如果在使用过程中，出现一些报错，建议将工作空间（jaka_robot）中的 build 文件夹和 devel 文件夹及 src 文件夹下的CMakeLists.txt 文件删除之后重新编译。
4. JAKA_ROS 开发实际上是通过 JAKA SDK 提供的函数接口控制机器人，相关接口的具体说明可以参考《JAKA C++二次开发用户手册》。

5. 因为同系列机器人 DH 参数一样，所以 C5/Pro5/Zu5 机器人的 ROS 配置包均可使用 Zu5配置包代替，C7/Zu7、Pro12/Zu12 同理。
6. jaka_driver 服务端和 moveit 服务端不能同时启动。

2.2 控制 JAKA 机器人

2.2.1 设置终端环境

在 ROS 中，每次打开一个新的终端窗口，都需要运行"source devel/setup.bash"命令来设置正确的环境变量，当开启多个终端窗口时有时会忘记添加环境变量从而导致某些错误的发生。为了解决这个问题，可以按照以下方法进行设置，以便在每次打开终端时自动加载 ROS 工作空间。

1. 在bash配置文件末尾添加以下信息，用来自动加载ROS工作空间，终端输入：

```
echo "source ~/xxx/devel/setup.sh" >> ~/.bashrc
```

注意：xxx是ros工作空间名称，这里我的工作空间存放在home目录下，名称为

jaka_robot_v2.2，所以输入echo "source ~/jaka_robot_v2.2/devel/setup.sh" >> ~/.bashrc，若是存放在其他位置注意修改路径。

2. 使配置生效，终端输入：

```
source ~/.bashrc
```

3. 确认是否写入成功，终端输入：

```
gedit ~/.bashrc
```

在打开的文件中看到如下最后一行则说明写入成功：

```
source /opt/ros/melodic/setup.bash|
export SVGA_VGPU10=0
export SVGA_VGPU10=0

source ~/jaka_robot_20230704/devel/setup.sh
source ~/jaka_robot_v2.2/devel/setup.sh
```

4. 重新编译，终端输入：catkin_make

之后每次新建终端就不再需要通过source ./devel/setup.bash来添加环境变量了。

```

/home/whm/jaka_robot_v2.2/src/jaka_zu3_moveit_config/launch/demo_gazebo.launch http:...
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~$ cd jaka_robot_v2.2/
whm@whm-virtual-machine:~/jaka_robot_v2.2$ roslaunch jaka_zu3_moveit_config demo_gazebo.launch
... logging to /home/whm/.ros/log/3f7d47dc-1fb5-11ee-938d-000c29cf701f/roslaunch-whm-virtual-machine-72505.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

WARN: unrecognized 'param' tag in <include> tag
WARN: unrecognized 'param' tag in <include> tag
started roslaunch server http://whm-virtual-machine:45957/

SUMMARY
=====

PARAMETERS
* /controller_list: [{'default': True...
* /gazebo/enable_ros_network: True
* /generic_hw_control_loop/cycle_time_error_threshold: 0.01
* /generic_hw_control_loop/loop_hz: 300
* /hardware_interface/joints: ['joint_1', 'join...
* /hardware_interface/sim_control_mode: 1
* /jaka_zu3_controller/action_monitor_rate: 1

```

注意：如果没有进行该设置，下面的操作每次打开一个终端都需要通过“source ./devel/setup.bash”命令来添加环境变量。

2.2.2 Jaka_Ros 驱动接口

实现功能：启动 Jaka_Ros 驱动服务端，通过 rosservice call 调用带参数的各功能服务。

具体 ROS 驱动接口通讯协议详见《jaka_driver_interface》。

测试机器人各功能服务的参考操作步骤如下：

1. 启动 robot_start_launch.launch 文件。在工作空间（jaka_robot_v2.2）右键打开一个终端，启动jaka_driver 服务端，通过 ros 命令传参告知服务端机器人IP：

```
roslaunch jaka_driver robot_start_launch.launch ip:=192.168.1.167
```

注意：ip:=xxx，xxx 为机器人 IP 地址，必填参数，否则连接不到机器人会报错。

```

/home/whm/jaka_robot_v2.2/src/jaka_driver/launch/robot_start_launch.launch http://local...
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ roslaunch jaka_driver robot_start_lau
nch.launch ip:=192.168.1.167
... logging to /home/whm/.ros/log/54ba62c0-34e0-11ee-9500-000c29cf701f/roslaunch
-whm-virtual-machine-51463.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://whm-virtual-machine:40281/

SUMMARY
=====

PARAMETERS
* /ip: 192.168.1.167
* /rostdistro: melodic
* /rosversion: 1.14.13

NODES
/
  jaka_driver (jaka_driver/jaka_driver)

auto-starting new master
process[master]: started with pid [51477]

```

2. 关节运动：使用 `rosservice call /jaka_driver/joint_move` 并按要求输入参数，控制机器人做关节运动。注：案例将运动接口设置为阻塞接口，若是需要改为非阻塞接口可以通过修改 `jaka_driver.cpp` 文件的相应接口参数实现。

```

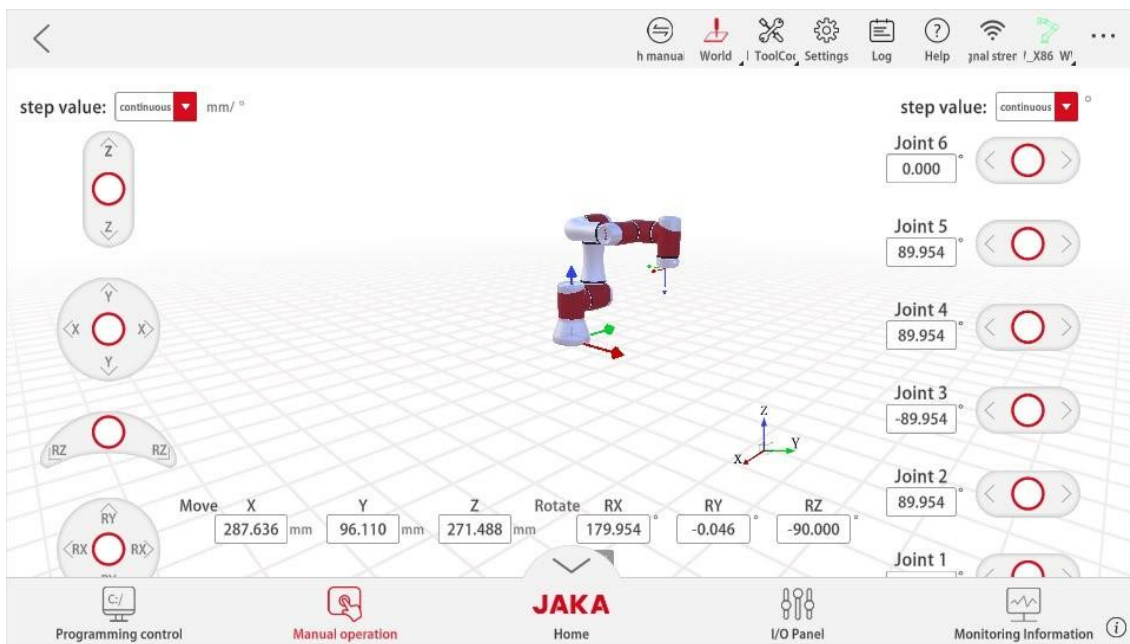
rosservice call /jaka_driver/joint_move "pose: [0,1.57,-1.57,1.57,1.57,0]
has_ref: false
ref_joint: [0]
mvvelo: 0.5
mvacc: 0.5
mvtime: 0.0
mvradii: 0.0
coord_mode: 0
index: 0"

```

```

whm@whm-virtual-machine: ~/jaka_robot_v2.2
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ rosservice call /jaka_driver/joint_move "pose: [0,1.57,-1.57,1.57,1.57,0]
has_ref: false
ref_joint: [0]
mvvelo: 0.5
mvacc: 0.5
mvtime: 0.0
mvradii: 0.0
coord_mode: 0
index: 0"
ret: 1
message: "joint_move has been executed"
whm@whm-virtual-machine:~/jaka_robot_v2.2$

```

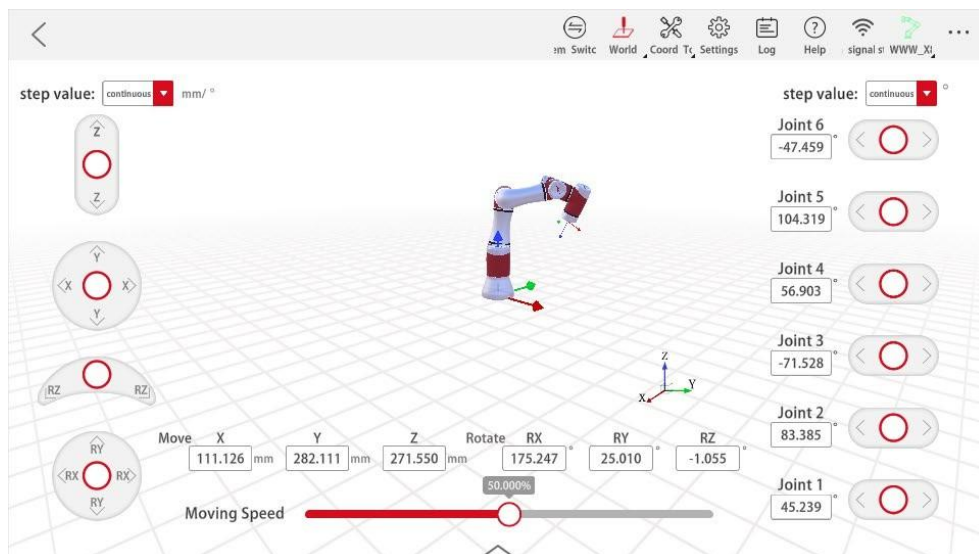


3. **直线运动**: 使用 `rosservice call /jaka_driver/linear_move` 并按要求输入参数，控制机器人运动。注意：以下示例 `pose` 参数仅作参考，具体参数需根据实际型号设置（该示例为 Zu3），否则可能导致执行失败，同时 APP 弹出“直线运动到目标位置失败”，失败原因：

- （1）超出运动范围；（2）遇到奇异点。

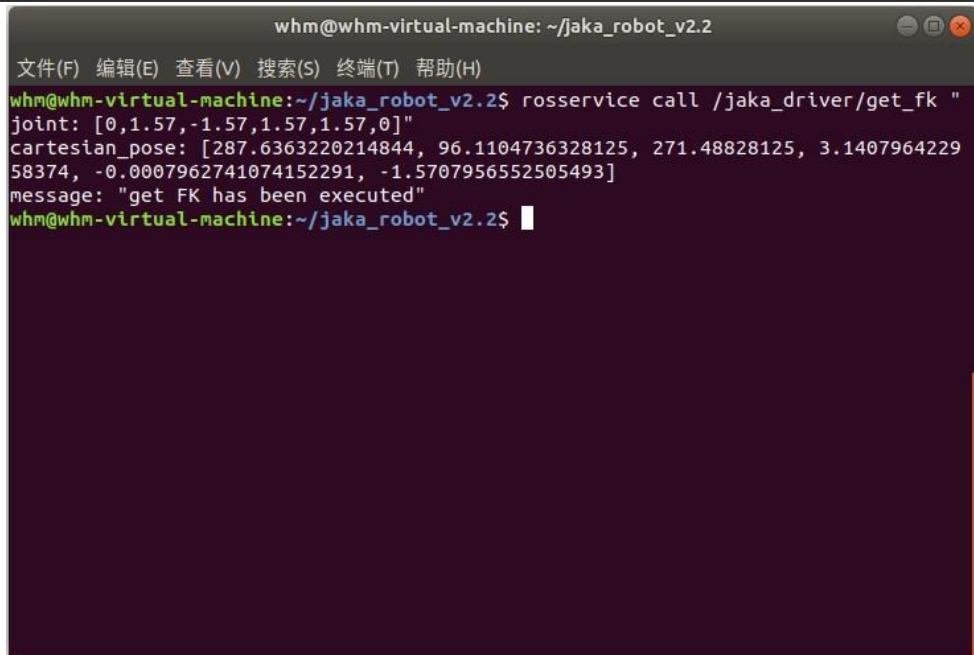

```
rosservice call /jaka_driver/linear_move "pose: [111.126,282.111,271.55,3.142,0,-0.698]
has_ref: false
ref_joint: [0]
mvvelo: 100
mvacc: 100
mvtime: 0.0
mvradii: 0.0
coord_mode: 0
index: 0"
```

```
whm@whm-virtual-machine: ~/jaka_robot_v2.2
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ rosservice call /jaka_driver/linear_m
ove "pose: [111.126,282.111,271.55,3.142,0,-0.698]
has_ref: false
ref_joint: [0]
mvvelo: 100
mvacc: 100
mvtime: 0.0
mvradii: 0.0
coord_mode: 0
index: 0"
ret: 1
message: "linear_move has been executed"
whm@whm-virtual-machine:~/jaka_robot_v2.2$
```



4. 求解机械臂正解：使用 `rosservice call /jaka_driver/get_fk` 并按要求输入参数，求取正解。

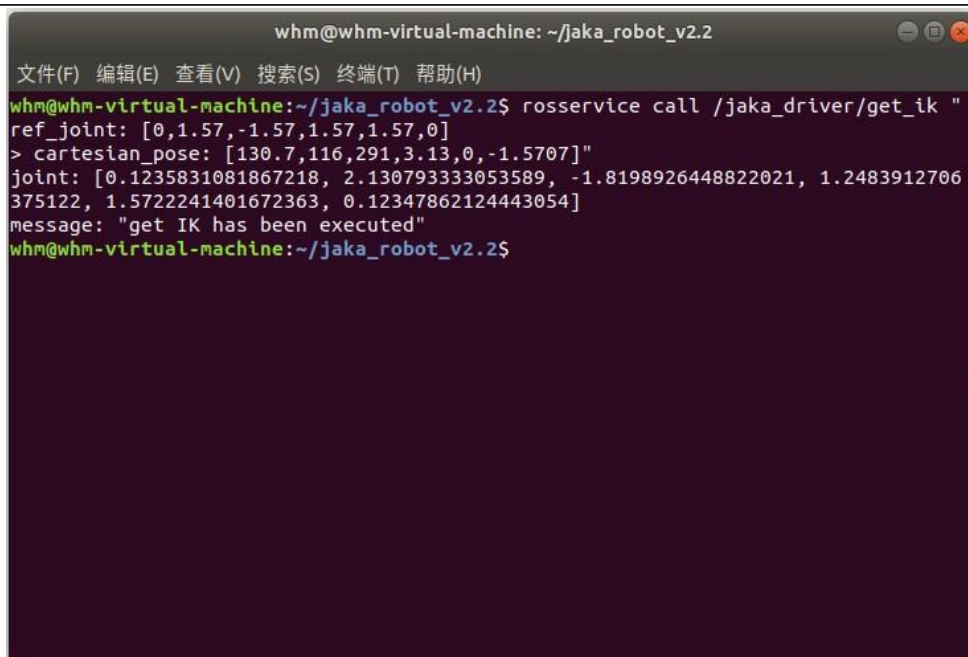
```
rosservice call /jaka_driver/get_fk "joint: [0,1.57,-1.57,1.57,1.57,0]"
```



```
whm@whm-virtual-machine: ~/jaka_robot_v2.2
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ rosservice call /jaka_driver/get_fk "
joint: [0,1.57,-1.57,1.57,1.57,0]"
cartesian_pose: [287.6363220214844, 96.1104736328125, 271.48828125, 3.1407964229
58374, -0.0007962741074152291, -1.5707956552505493]
message: "get FK has been executed"
whm@whm-virtual-machine:~/jaka_robot_v2.2$
```

5. 求解机械臂逆解：使用 `rosservice call /jaka_driver/get_ik` 并按要求输入参数，求取逆解。

```
rosservice call /jaka_driver/get_ik "ref_joint: [0,1.57,-1.57,1.57,1.57,0]
cartesian_pose: [130.7,116,291,3.13,0,-1.5707]"
```



```
whm@whm-virtual-machine: ~/jaka_robot_v2.2
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ rosservice call /jaka_driver/get_ik "
ref_joint: [0,1.57,-1.57,1.57,1.57,0]
> cartesian_pose: [130.7,116,291,3.13,0,-1.5707]"
joint: [0.1235831081867218, 2.130793333053589, -1.8198926448822021, 1.2483912706
375122, 1.5722241401672363, 0.12347862124443054]
message: "get IK has been executed"
whm@whm-virtual-machine:~/jaka_robot_v2.2$
```

2.2.3 Moveit 和 Gazebo 联合使用

实现功能：启动 Rviz 和 Gazebo，在 Rviz 中规划路径并执行，在 Gazebo 中的机器人仿真模型会移动到相应位置。具体操作步骤为：

1. 启动 Zu3 的 demo_gazebo.launch 文件同时启动 Rviz 和 Gazebo。若要启动其它型号机器人修改输入命令为相应 moveit 配置功能包（jaka_xx_moveit_config）即可。

```
roslaunch jaka_zu3_moveit_config demo_gazebo.launch
```

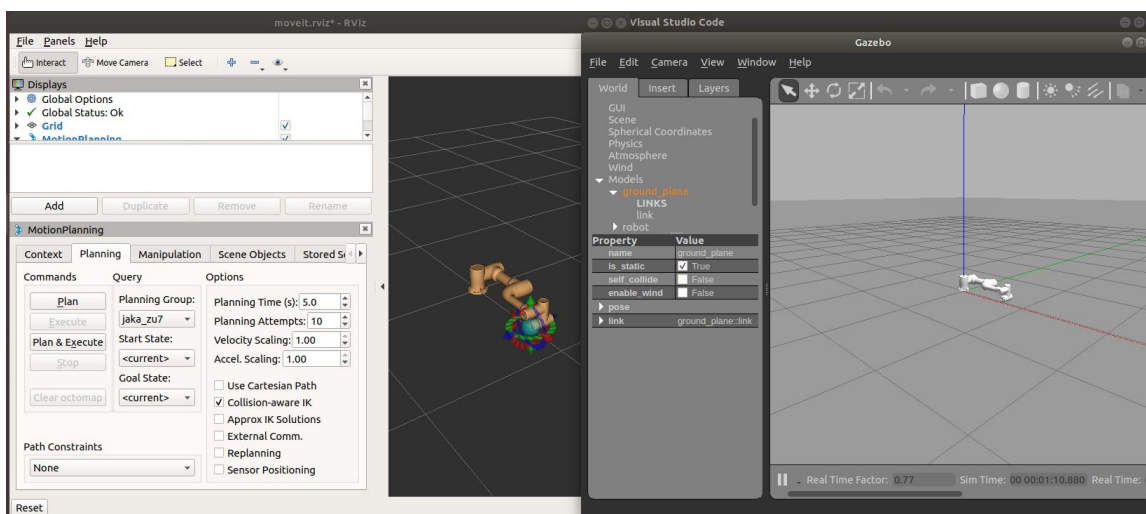
```
/home/whm/jaka_robot_v2.2/src/jaka_zu3_moveit_config/launch/demo_gazebo.launch http://...
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ roslaunch jaka_zu3_moveit_config demo_gazebo.launch
... logging to /home/whm/.ros/log/add9125a-34f1-11ee-9500-000c29cf701f/roslaunch
-whm-virtual-machine-90562.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

WARN: unrecognized 'param' tag in <include> tag
WARN: unrecognized 'param' tag in <include> tag
started roslaunch server http://whm-virtual-machine:38103/

SUMMARY
=====

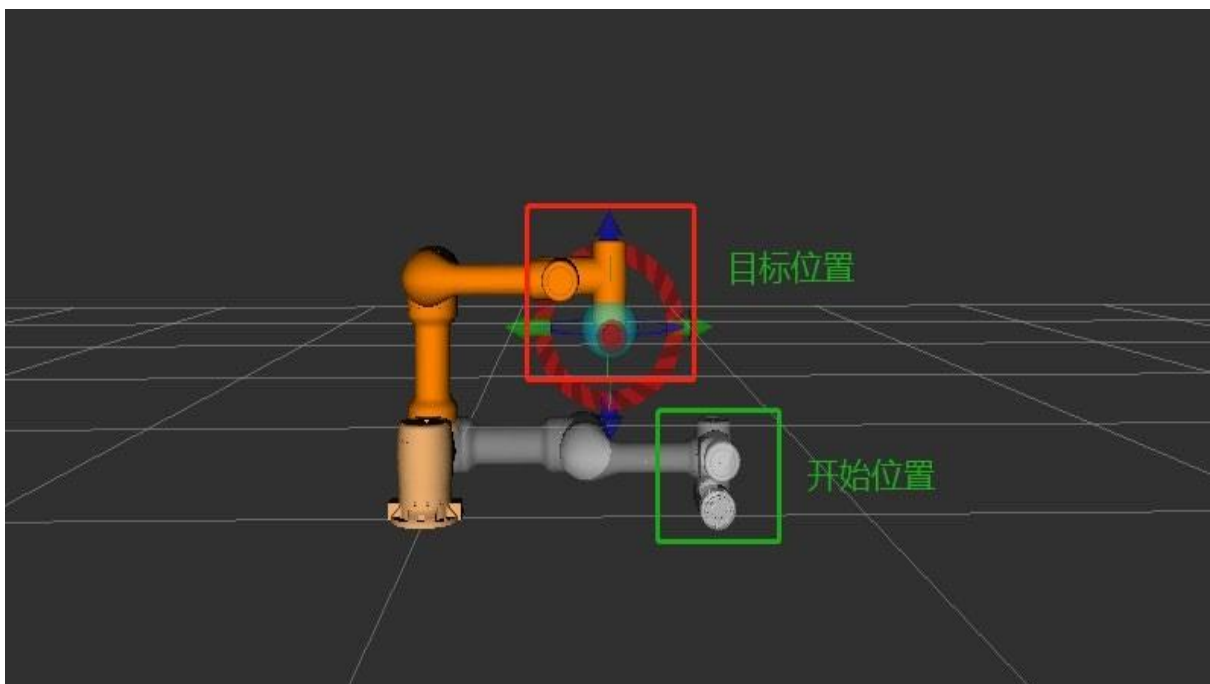
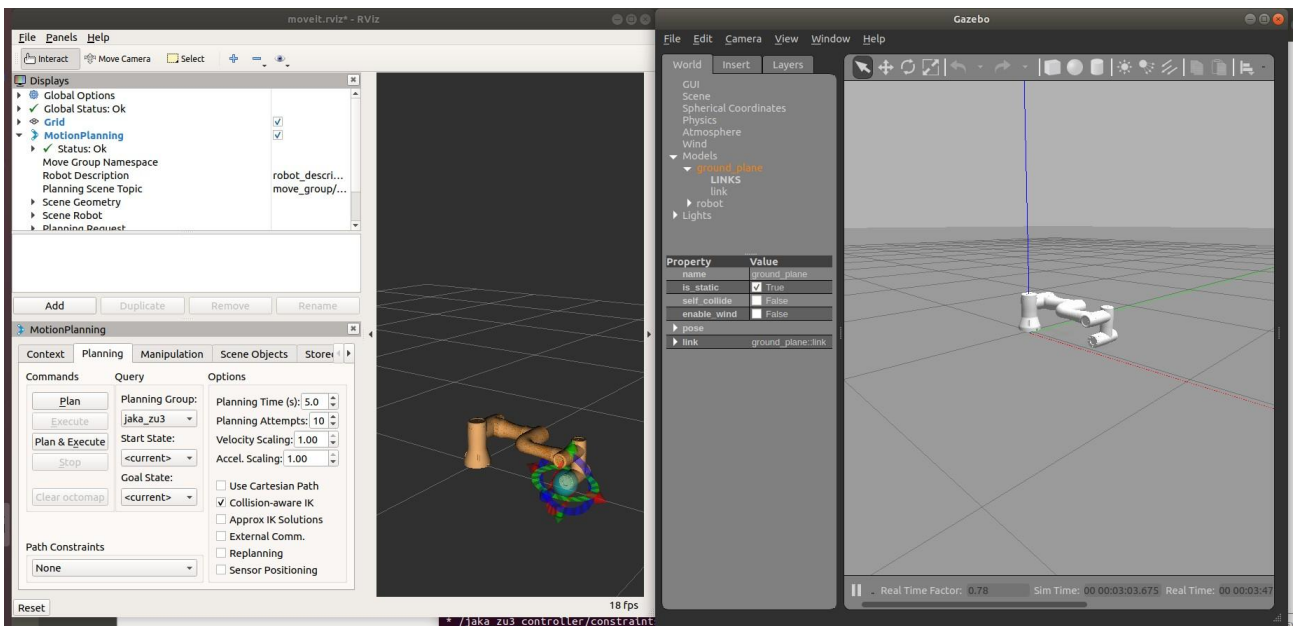
PARAMETERS
* /controller_list: [{'default': True...
* /gazebo/enable_ros_network: True
* /generic_hw_control_loop/cycle_time_error_threshold: 0.01
* /generic_hw_control_loop/loop_hz: 300
* /hardware_interface/joints: ['joint_1', 'join...
* /hardware_interface/sim_control_mode: 1
* /jaka_zu3_controller/action_monitor_rate: 1
* /jaka_zu3_controller/constraints/goal_time: 0.6
```

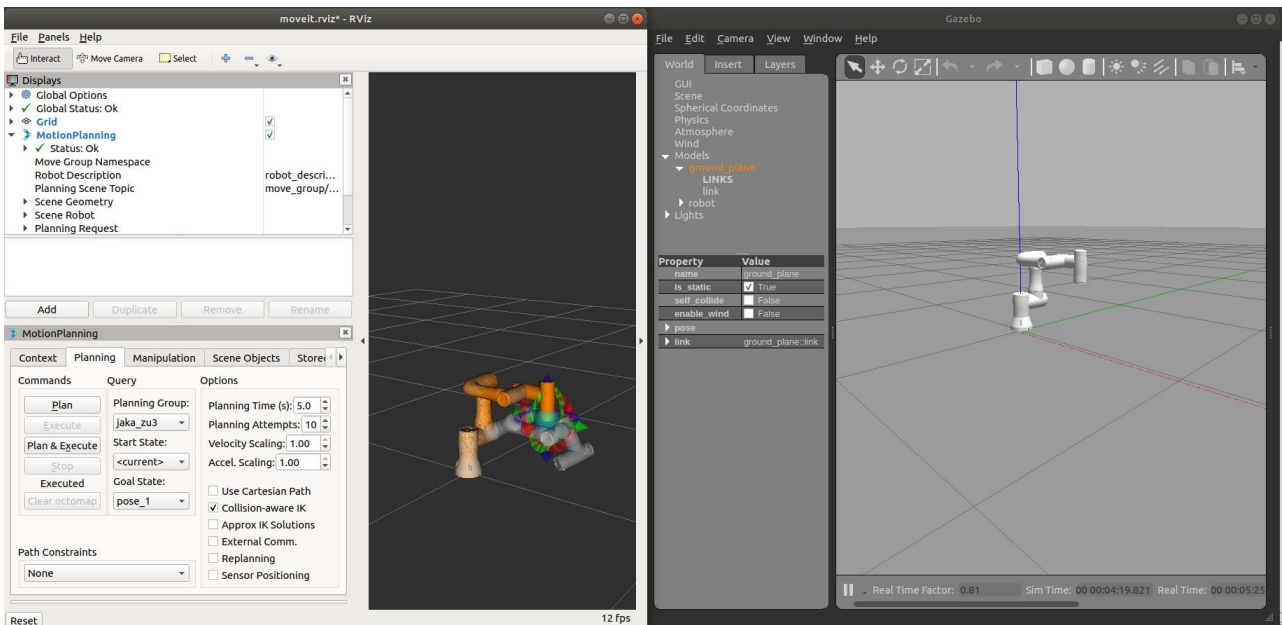
生成如下界面：



2. 从 RVIZ 界面的“Goal state”选择一个目标位置，点击“Plan & Execute”，RVIZ 界面

会显示机器人的轨迹，并驱动 Gazebo 中仿真机器人到设置的目标位置。





注意：启动demo_gazebo.launch 文件会出现报错 “No p gain specified for pid.”，该报错是由于我们 Moveit 设置使用的控制器是位置控制器，pid 不需要设置，就会存在这个报错，但不会影响使用。

```

/home/whm/jaka_robot_v2.2/src/jaka_zu3_moveit_config/launch/demo_gazebo.launch http://...
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

in namespace: /
[spawn_gazebo_model-4] process has finished cleanly
log file: /home/whm/.ros/log/add9125a-34f1-11ee-9500-000c29cf701f/spawn_gazebo_model-4*.log
[ INFO] [1691392313.298161782, 0.193000000]: gazebo_ros_control plugin is waiting for model URDF in parameter [robot_description] on the ROS param server.
[ERROR] [1691392313.412070997, 0.193000000]: No p gain specified for pid. Name space: /gazebo_ros_control/pid_gains/joint_1
[ERROR] [1691392313.412788924, 0.193000000]: No p gain specified for pid. Name space: /gazebo_ros_control/pid_gains/joint_2
[ERROR] [1691392313.413230377, 0.193000000]: No p gain specified for pid. Name space: /gazebo_ros_control/pid_gains/joint_3
[ERROR] [1691392313.414224713, 0.193000000]: No p gain specified for pid. Name space: /gazebo_ros_control/pid_gains/joint_4
[ERROR] [1691392313.414863324, 0.193000000]: No p gain specified for pid. Name space: /gazebo_ros_control/pid_gains/joint_5
[ERROR] [1691392313.415350758, 0.193000000]: No p gain specified for pid. Name space: /gazebo_ros_control/pid_gains/joint_6
[ INFO] [1691392313.422103905, 0.193000000]: Loaded gazebo_ros_control.
[INFO] [1691392313.549248, 0.325000]: Controller Spawner: Waiting for service controller_manager/switch_controller
[INFO] [1691392313.554263, 0.325000]: Controller Spawner: Waiting for service controller_manager/unload_controller
[INFO] [1691392313.559908, 0.325000]: Loading controller: joint_state_controller

```

2.2.4 Moveit 和真实机器人联合使用

实现功能：启动 moveit 服务端及 RVIZ，RVIZ 规划路径，真实机械臂会移动到相应位置。

节卡机器人股份有限公司 JAKA Robotics Co., Ltd.

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Road, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

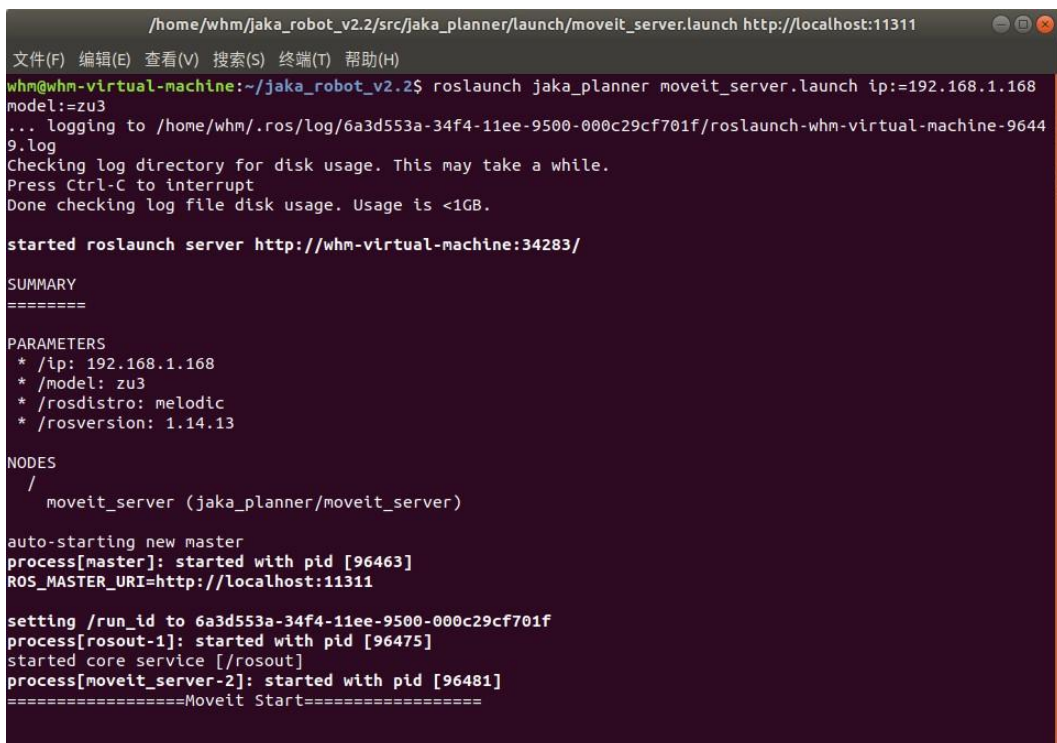
前提条件：jaka_driver 服务端和 Moveit 服务端不能同时启动，所以启动 moveit 服务端之前确认jaka_driver 服务端已经关闭（即未启动 robot_start_launch.launch）。

具体操作步骤为：

1. 启动 moveit_server.launch 文件(Moveit 服务端)。启动 moveit 服务端时通过 ros 命令传参告知服务端机器人型号及 IP。新建终端，并输入：

```
roslaunch jaka_planner moveit_server.launch ip:=192.168.1.168 model:=zu3
```

注意：ip:=xxx 中，xxx 表示机器人 IP 地址。model:=xxx 中，xxx 表示各机器人型号，均为小写，例如：zu3、minicobo、pro16。



```
/home/whm/jaka_robot_v2.2/src/jaka_planner/launch/moveit_server.launch http://localhost:11311
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ roslaunch jaka_planner moveit_server.launch ip:=192.168.1.168
model:=zu3
... logging to /home/whm/.ros/log/6a3d553a-34f4-11ee-9500-000c29cf701f/roslaunch-whm-virtual-machine-9644
9.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://whm-virtual-machine:34283/

SUMMARY
=====

PARAMETERS
* /ip: 192.168.1.168
* /model: zu3
* /rostdistro: melodic
* /rosversion: 1.14.13

NODES
/
  moveit_server (jaka_planner/moveit_server)

auto-starting new master
process[master]: started with pid [96463]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 6a3d553a-34f4-11ee-9500-000c29cf701f
process[rosout-1]: started with pid [96475]
started core service [/rosout]
process[moveit_server-2]: started with pid [96481]
=====Moveit Start=====
```

2. 另外新建一个终端，启动 demo.launch 文件（Moveit 客户端）。

```
roslaunch jaka_zu3_moveit_config demo.launch
```



```

/home/whm/jaka_robot_v2.2/src/jaka_zu3_moveit_config/launch/demo.launch http://localh...
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
whm@whm-virtual-machine:~/jaka_robot_v2.2$ roslaunch jaka_zu3_moveit_config demo
.launch
... logging to /home/whm/.ros/log/6a3d553a-34f4-11ee-9500-000c29cf701f/roslaunch
-whm-virtual-machine-100662.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

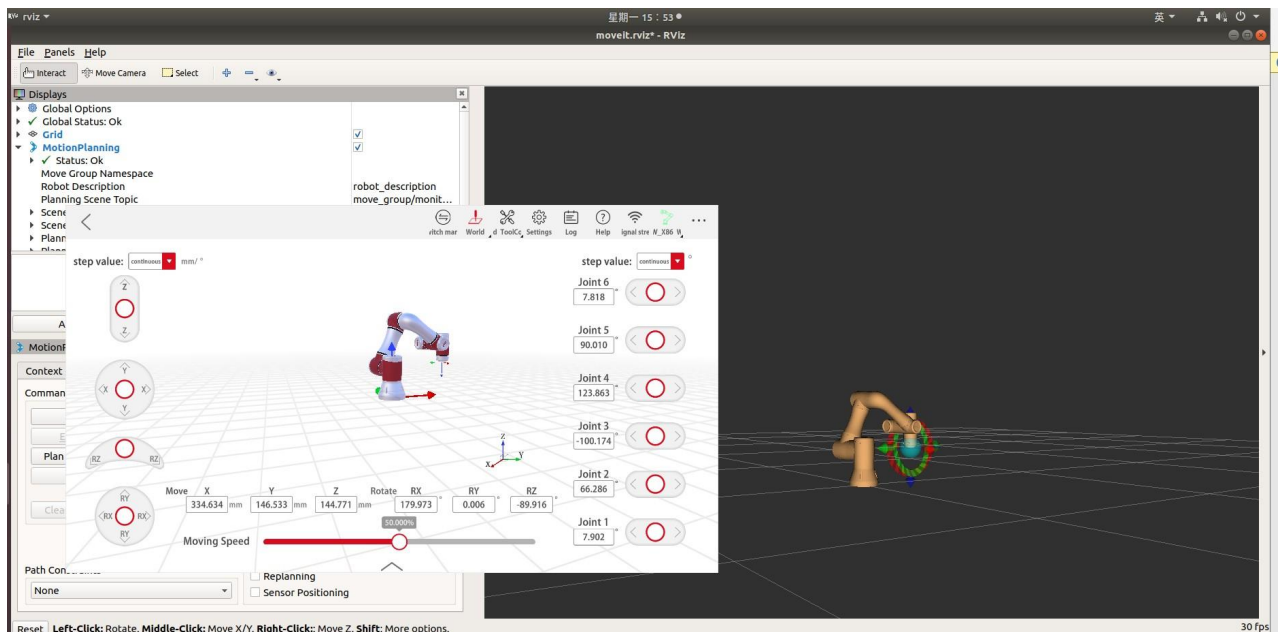
WARN: unrecognized 'param' tag in <include> tag
WARN: unrecognized 'param' tag in <include> tag
started roslaunch server http://whm-virtual-machine:38005/

SUMMARY
=====

PARAMETERS
* /joint_state_publisher/source_list: ['/joint_states']
* /move_group/allow_trajectory_execution: True
* /move_group/capabilities:
* /move_group/controller_list: [{'default': True...
* /move_group/disable_capabilities:
* /move_group/generic_hw_control_loop/cycle_time_error_threshold: 0.01
* /move_group/generic_hw_control_loop/loop_hz: 300
* /move_group/hardware_interface/joints: ['joint_1', 'join...

```

生成如下界面：Rviz 显示的姿态和实体机器人当前姿态一致。



3. 从 RVIZ 界面的“Goal state”选择一个目标位置，点击“Plan & Execute”，RVIZ 界面会显示机器人的轨迹，并驱动实体机器人到设置的目标位置。


```

/home/whm/jaka_robot_v2.2/src/jaka_zu3_moveit_config/launch/demo.launch http://localhost:11311
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
in datastructure
[ INFO] [1691472740.755602409]: jaka_zu3[RRT]: Starting planning with 1 states already
in datastructure
[ INFO] [1691472740.756695880]: jaka_zu3[RRT]: Created 8 states
[ INFO] [1691472740.757805578]: jaka_zu3[RRT]: Created 14 states
[ INFO] [1691472740.757950866]: jaka_zu3[RRT]: Created 12 states
[ INFO] [1691472740.759581143]: jaka_zu3[RRT]: Created 19 states
[ INFO] [1691472740.759820398]: ParallelPlan::solve(): Solution found by one or more t
hreads in 0.004734 seconds
[ INFO] [1691472740.760213667]: jaka_zu3[RRT]: Starting planning with 1 states already
in datastructure
[ INFO] [1691472740.760352766]: jaka_zu3[RRT]: Starting planning with 1 states already
in datastructure
[ INFO] [1691472740.761471861]: jaka_zu3[RRT]: Created 8 states
[ INFO] [1691472740.769010685]: jaka_zu3[RRT]: Created 32 states
[ INFO] [1691472740.769711384]: ParallelPlan::solve(): Solution found by one or more t
hreads in 0.009617 seconds
[ INFO] [1691472740.769852707]: SimpleSetup: Path simplification took 0.000009 seconds
and changed from 2 to 2 states
[ INFO] [1691472745.043881399]: Controller jaka_zu3_controller successfully finished
[ INFO] [1691472745.062659001]: Completed trajectory execution with status SUCCEEDED .
..
[ WARN] [1691472746.147489713]: Maybe failed to update robot state, time diff: 0.067s

```

4. 代码端控制：请参考以下文件

`/jaka_robot_v2.2/src/jaka_planner/src/moveit_test.cpp`

注：（1）示例代码型号对应的是 zu3，若机器人为其他型号需要先确认代码中的位置是否可达且在安全区域。

（2）启动方式：分别在 3 个终端输入以下命令

```

roslaunch jaka_planner moveit_server.launch ip:=192.168.1.168 model:=zu3
roslaunch jaka_zu3_moveit_config demo.launch
roslaunch jaka_planner moveit_test

```

第 3 章 JAKA 机器人 ROS 功能包应用案例

案例及使用说明文档下载地址：

https://github.com/JakaCobot/jaka_robot