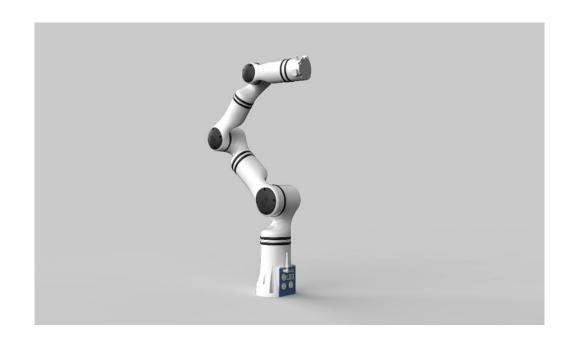


睿尔曼机器人 rm_control 使用说明书 V1.1



睿尔曼智能科技(北京)有限公司



文件修订记录:

版本号	时间	备注
V1.0	2024-1-16	拟制
V1.1	2024-7-4	修订(添加 gen72 相关文件)



目录

1. rm_control 功能包说明	 3
2. rm_control 功能包使用	3
2.1 功能包基础使用	3
2.2 功能包进阶使用	 4
3. rm_control 功能包架构说明	 4
3.1 功能包文件总览	4
4. rm control 话题说明	4



1. rm control 功能包说明

rm_control 功能包为实现 moveit 控制真实机械臂时所必须的一个功能包,该功能包的主要作用为将 moveit 规划好的路径点进行进一步的细分,将细分后的路径点以透传的方式给到 rm_driver,实现机械臂的规划运行。

- 1. 功能包使用。
- 2. 功能包架构说明。
- 3. 功能包话题说明。

通过这三部分内容的介绍可以帮助大家:

- 1. 了解该功能包的使用。
- 2. 熟悉功能包中的文件构成及作用。
- 3. 熟悉功能包相关的话题,方便开发和使用

代码链接: https://github.com/RealManRobot/rm robot/tree/main/rm control。

2. rm control 功能包使用

2.1 功能包基础使用

首先配置好环境完成连接后我们可以通过以下命令直接启动节点,运行 rm_control 功能包。

```
rm@rm-desktop:~$ roslaunch rm_control rm_<arm_type>_control.launch
```

在实际使用时需要将以上的<arm_type>更换为实际的机械臂型号,可选择的机械臂型号有65、63、eco65、75、gen72。

例如 65 机械臂的启动命令:

```
rm@rm-desktop:~$ roslaunch rm_control rm_65_control.launch
```

节点启动成功后,将显示以下画面。

```
NODES
/
rm_control (rm_control/rm_control)

ROS_MASTER_URI=http://192.168.0.244:11311

process[rm_control-1]: started with pid [10413]
[ INFO] [1705395555.850722127]: arm_dof6 = 6
```

在单独启动该功能包的节点时并不发挥作用,需要结合 rm_driver 功能包和 moveit 的相关节点一起使用才能发挥作用,详细请查看《rm moveit config 详解》相关内容。



2.2 功能包进阶使用

在 rm_control 功能包中也有一些参数可以进行配置,可以直接在 launch 文件中进行了解查看。

当前该文件只有一个配置参数。

Arm_Dof: 代表机械臂的自由度, 当为 RM65、RML63、ECO65 时应配置为 6, 当为 RM75 时应配置为 7。

3. rm control 功能包架构说明

3.1 功能包文件总览

当前 rm_driver 功能包的文件构成如下。

```
#编译规则文件
CMakeLists.txt
launch
                           #63 启动文件
  rm_63_control.launch
                           #65 启动文件
 - rm_65_control.launch
 - rm_75_control.launch
                          #75 启动文件
  rm eco65 control.launch
                          #eco65 启动文件
 rm_gen72_control.launch
                          #gen72 启动文件
                          #依赖声明文件
package.xml
src
  - cubicSpline.h
                           #三次样条插值头文件
                           #代码源文件
  - rm_control.cpp
```

4. rm_control 话题说明

如下为该功能包的话题说明。

```
/joint_states
/rm_driver/ArmError
/rm_driver/Clear_System_Err
/rm_driver/JointPos
/rm_group/follow_joint_trajectory/cancel
/rm_group/follow_joint_trajectory/feedback
/rm_group/follow_joint_trajectory/goal
/rm_group/follow_joint_trajectory/result
/rm_group/follow_joint_trajectory/status
/rosout
/rosout_agg
```



我们主要关注以下几个话题。

Publishers:代表其当前发布的话题,其最主要发布的话题为/rm_driver/JointPos,我们通过该话题将细分后的点发布给 rm_driver 节点, rm_driver 节点再通过透传的指令方式给到机械臂执行相对应的路径。

Action Servers:代表其接受和发布的动作信息,rm_group/follow_joint_trajectory 动作为 rm_control 与 moveit 进行通信的桥梁,通过该动作 rm_control 接收到 moveit 规划的路径,rm_control 会将这些路径进行进一步细分由以上话题给到 rm_driver。

剩余话题和服务使用场景较少,这里不做详细介绍,大家可自行了解。