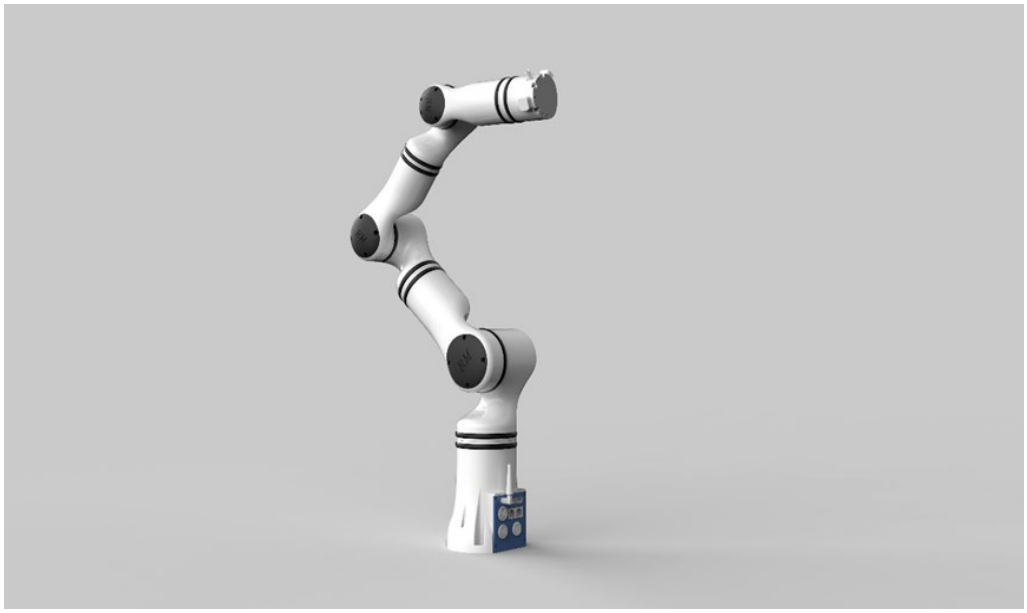
睿尔曼机器人 rm_control 使用说明书 V1.0



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	2023-11-22	拟制

目录

1. rm_control 功能包说明

rm_control 功能包为实现 moveit2 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 moveit2 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，在下文中将通过以下几个方面详细介绍该功能包。

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

源码地址：https://github.com/RealManRobot/ros2_rm_robot.git。

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:~$ ros2 launch    rm_control rm_<arm_type>_control.launch.py
```

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。

例如 65 机械臂的启动命令：

```
rm@rm-desktop:~$ ros2 launch    rm_control rm_65_control.launch.py
```

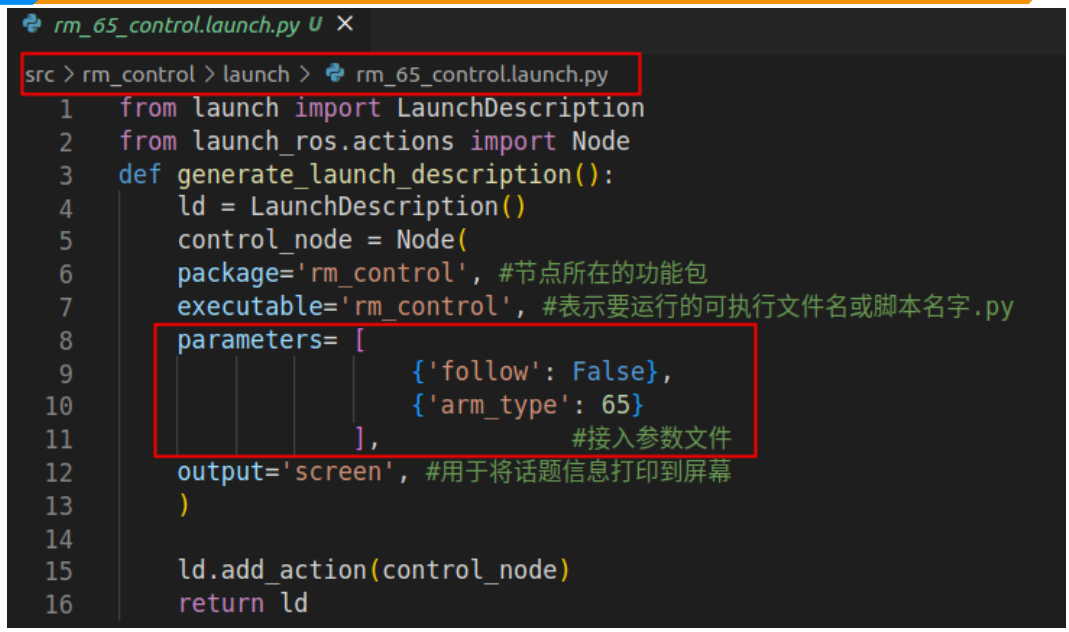
节点启动成功后，将显示以下画面。

```
[INFO] [launch]: All log files can be found below /home/rm/.ros/log/2023-11-22-14-00-34-107955-rm-desktop-13727
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [rm_control-1]: process started with pid [13728]
```

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 moveit2 的相关节点一起使用才能发挥作用，详细请查看《rm_moveit2_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，由于参数并不是很多，这边将参数直接在 launch 文件中进行了配置。



如上图所示第一个红框框出的位置为文件的路径，第二个框出的位置为当前可配置的参数。

参数 follow：代表当前透传使用的跟随模式，true:高跟随，false:低跟随。高跟随即机械臂运动方式与透传完全一致，需要根据透传的速率和机械臂的速度、加速度参数进行较详细的计算，使用门槛较高，但控制精细。低跟随即机械臂会基本根据透传速率和速度、加速度向透传点运动，若有来不及到达的点可能会有丢弃现象发生，使用门槛低，控制不太精细，但基本满足使用。

参数 arm_type：代表当前使用的机械臂型号，可以选择的参数有 65（65 系列）、651（eco65）、632（63 系列）、75（75 系列）。

再实际使用时，我们选择对应的 launch 文件启动时会自动选择正确的型号，若有特殊要求可在此处进行相应的参数修改，修改之后需要在工作空间目录下进行重新编译，之后修改的配置才会生效。

在工作空间目录运行 colcon build 指令。

```
rm@rm-desktop: ~/ros2_ws$ colcon build
```

编译成功后可按如上指令进行功能包启动。

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。

```
├── CMakeLists.txt                    #编译规则文件
```

```
├── include                          #依赖头文件文件夹
│   ├── cubicSpline.h                #三次样条插值头文件
│   └── rm_control.h                 #rm_control 头文件
├── launch
│   ├── rm_63_control.launch.py      #63 启动文件
│   ├── rm_65_control.launch.py      #65 启动文件
│   ├── rm_75_control.launch.py      #75 启动文件
│   └── rm_eco65_control.launch.py   #eco65 启动文件
├── package.xml                      #依赖声明文件
└── src
    └── rm_control.cpp               #代码源文件
```

4. rm_control 话题说明

如下为该功能包的话题说明。

```
Subscribers:
  /parameter_events: rcl_interfaces/msg/ParameterEvent
Publishers:
  /parameter_events: rcl_interfaces/msg/ParameterEvent
  /rm_driver/movej_canfd_cmd: rm_ros_interfaces/msg/Jointpos
  /rosout: rcl_interfaces/msg/Log
Service Servers:
  /rm_control/describe_parameters: rcl_interfaces/srv/DescribeParameters
  /rm_control/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
```

```
    /rm_control/get_parameters: rcl_interfaces/srv/GetParameters

    /rm_control/list_parameters: rcl_interfaces/srv/ListParameters

    /rm_control/set_parameters: rcl_interfaces/srv/SetParameters

    /rm_control/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically

  Service Clients:



  Action Servers:

    /rm_group_controller/follow_joint_trajectory:
control_msgs/action/FollowJointTrajectory

  Action Clients:
```

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/movej_canfd_cmd，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_controller/follow_joint_trajectory 动作为 rm_control 与 moveit2 进行通信的桥梁，通过该动作 rm_control 接收到 moveit2 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。