



睿尔曼机械臂新增话题教程 (ROS1) V1.0.0



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
V1.0.0	2025-12-5	拟制



目录

1. 简介.....	4
2. ROS1 增加话题教程.....	4
2.1 获取 JSON.....	4
2.2 增加自定义 msg 文件.....	4
2.3 创建订阅器.....	5
2.4 发布者发布运行结果.....	6
2.5 验证.....	8



1. 简介

当前 ROS1 功能包并未完全涵盖控制器所有功能。为此，睿尔曼提供了新增话题的教程，以便客户有功能包之外的紧急需求时，可自行增加目标话题。

2. ROS1 增加话题教程

ROS1 增加新话题的步骤可以简单描述为：

- 1) 在官网找到目标功能的 JSON 协议指令；
- 2) 增加自定义 msg，用于输入数据；
- 3) 创建订阅器，将输入数据拼接成 JSON 协议指令发送给控制器；
- 4) 创建发布者，解析 JSON 返回数据以检测指令运行情况。

下面，以增加“扩展关节速度环控制”功能为例进行说明。

2.1 获取 JSON

根据官网 JSON 协议：[控制器扩展设备指令集（选配） | 睿尔曼智能科技](#)，得到 JSON 协议指令。

```
{"command": "expand_set_speed", "speed": -50}
```

```
{"command": "expand_set_speed", "set_speed_state": true}
```

其中，“扩展关节速度环控制”指令为 `expand_set_speed`，参数 `speed`（有符号整型）表示扩展关节速度；执行状态由布尔字段 `set_speed_state` 返回。

2.2 增加自定义 msg 文件

在 `ros_ws/src/rm_msgs/msg` 文件夹增加 `Expand_Speed.msg` 文件。

```
Expand_Speed.msg
rm_msgs > msg > Expand_Speed.msg
1  int16 speed          #扩展关节速度环控制速度百分比， -100~100
```

打开 `ros_ws/src/rm_msgs/CMakeLists.txt`，在 `add_message_files()`中追



加自定义消息 `Expand_Speed.msg`，用于扩展关节速度指令的接口定义。

```
#文件路径 ros_ws/src/rm_msgs/CMakeLists.txt
add_message_files(
#一些已有 msg 文件
Expand_Speed.msg
)
```

新建终端，运行命令行。

```
catkin build rm_msgs
source ~/ros_ws/devel/setup.bash
```

在文件 `ros_ws/src/rm_driver/src/rm_robot.h` 中增加对应头文件

```
#include <rm_msgs/Expand_Speed.h>
```

2.3 创建订阅器

创建订阅器，将输入数据拼接成指令发送给控制器：

```
{"command": "expand_set_speed", "speed": -50}.
```

涉及改动的文件：`ros_ws/src/rm_driver/src/rm_driver.cpp`、`ros_ws/src/rm_driver/src/rm_robot.h`。

在 `rm_robot.h` 文件中声明订阅器及发布者。

```
rm_robot.h x  rm_driver.cpp
rm_driver > src > C rm_robot.h > ...
680 ros::Publisher pub_getExpandState_Result;
681 /*****扩展关节速度环控制*****/
682 ros::Subscriber sub_setExpandSpeed_Cmd;
683 ros::Publisher pub_setExpandSpeed_Result;
684 /*****扩展关节位置环控制*****/
```

在 `rm_driver.cpp` 文件中注册订阅/发布话题。

```
rm_robot.h  rm_driver.cpp x
rm_driver > src > C rm_driver.cpp > main(int, char **)
2373 int main(int argc, char **argv)
2907 pub_getExpandState_Result = nh.advertise<rm_msgs::ExpandState>("/rm_driver/Get_Expand_State_Result", 10);
2908 /*****扩展关节速度环控制*****/
2909 sub_setExpandSpeed_Cmd = nh.subscribe("/rm_driver/Set_Expand_Speed_Cmd", 10, SetExpandSpeed_Callback);
2910 pub_setExpandSpeed_Result = nh.advertise<std_msgs::Bool>("/rm_driver/Set_Expand_Speed_Result", 10);
```

订阅器订阅 `rm_msgs::Expand_Speed` 数据，拼接 JSON 控制指令发送给控制器。



```
rm_robot.h  rm_driver.cpp x
rm_driver > src > rm_driver.cpp > SetExpandSpeed_Callback(const rm_msgs::Expand_Speed)
2197     }
2198     void SetExpandSpeed_Callback(const rm_msgs::Expand_Speed msg)
2199     {
2200         int res=0;
2201         res = Set_Expand_Speed_Cmd(msg.speed);
2202         if(res==0)
2203         {
2204             ROS_INFO("Set_Expand_Speed_Cmd success!\n");
2205         }
2206         else
2207         {
2208             ROS_INFO("Set_Expand_Speed_Cmd failed!\n");
2209         }
2210     }
```

```
rm_robot.h x  rm_driver.cpp
rm_driver > src > rm_robot.h > Set_Expand_Pos_Cmd(int, int)
3397     }
3398     int Set_Expand_Speed_Cmd(int speed)
3399     {
3400         cJSON *root;
3401         char *data;
3402         char buffer[200];
3403         int res;
3404         //创建根节点对象
3405         root = cJSON_CreateObject();
3406         //加入字符串对象
3407         cJSON_AddStringToObject(root, "command", "expand_set_speed");
3408         cJSON_AddNumberToObject(root, "speed", speed);
3409         data = cJSON_Print(root);
3410         sprintf(buffer, "%s\r\n", data);
3411         // ROS_INFO("get_rm_plus_touch command: %s",buffer);
3412         res = package_send(Arm_Socket, buffer, strlen(buffer), 0);
3413         cJSON_Delete(root);
3414         free(data);
3415         if (res < 0)
3416         {
3417             return 1;
3418         }
3419         return 0;
3420     }
```

2.4 发布者发布运行结果

解析 JSON 返回数据，并利用 2.3 中已创建的发布者发布布尔型运行结果。

在 rm_robot.h 文件的 Parser_Msg 函数体中进行 JSON 返回数据的解析。



```
C rm_robot.h x  G+ rm_driver.cpp
rm_driver > src > C rm_robot.h > Parser_Msg(char *)
7484 int Parser_Msg(char *msg)
7930 if (json_state != NULL)
8305 else if(!strcmp("expand_set_speed", json_state->valuelstring)) //扩展关节速度控制返回
8307 {
8308     json_state = cJSON_GetObjectItem(root, "set_speed_state");
8308     if (json_state != NULL)
8309     {
8310         if (json_state->type == cJSON_True)
8311         {
8312             RM_Joint.state = true;
8313             return SET_EXPAND_SPEED;
8314         }
8315         else if(json_state->type == cJSON_False)
8316         {
8317             RM_Joint.state = false;
8318             return SET_EXPAND_SPEED;
8319         }
8320     }
8321 }
```

对于简单的布尔型返回值，可以直接复用已有的变量 RM_Joint.state。在 rm_robot.h 中增加宏 SET_EXPAND_SPEED。

```
C rm_robot.h x  G+ rm_driver.cpp
rm_driver > src > C rm_robot.h > SET_EXPAND_SPEED
810 #define SET_ARM_CONTINUE 0X66
811 #define EXPAND_CURRENT_STATE 0X67
812 #define SET_EXPAND_SPEED 0X68
813 #define EXPAND_IN_POSITION 0X69
```

在 rm_driver.cpp 的 main 函数里，增加如下代码片段，发布运行结果。

```
C rm_robot.h  G+ rm_driver.cpp x
rm_driver > src > G+ rm_driver.cpp > main(int, char **)
2373 int main(int argc, char **argv)
2940 while (ros::ok())
2942     while (realtime_arm_joint_state == true)
2960         if ((temp[0] == 0x0A) && (buffer_cnt > 1))
2972             if (msg == 0x0D)
3394                 break;
3395             case SET_EXPAND_SPEED:
3396                 state.data = RM_Joint.state;
3397                 pub_setExpandSpeed_Result.publish(state);
3398                 break;
```

此外，若发布数据为自定义 msg，可参考“扩展关节状态获取”的处理方式：创建 EXPAND_STATE Expand_Current_State 暂存 JSON 解析结果，再将其逐项赋于 rm_msgs::ExpandState expandState 并通过对应发布者发出。详情可参考 ROS2.6.0 对应代码。



2.5 验证

新建三个终端。

终端一：编译功能包。若无报错，启动对应型号的驱动功能包。

```
cd ~/ros_ws
catkin build
source ~/ros_ws/devel/setup.bash
roslaunch rm_driver rm_65_driver.launch #以 65 为例
```

终端二：输入以下命令并回车，监听节点每次回传的成功/失败状态。

```
rostopic echo /rm_driver/Set_Expand_Speed_Result
```

终端三：输入以下命令并回车，向订阅器发布参数。

```
rostopic pub /rm_driver/Set_Expand_Speed_Cmd rm_msgs/Expand_Speed "speed:
50"
```

若成功添加，将注意到扩展关节以目标速度转动，并且终端二返回 True。