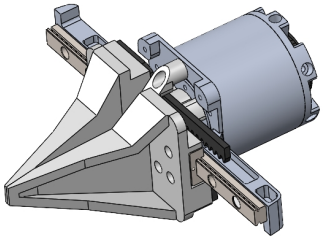
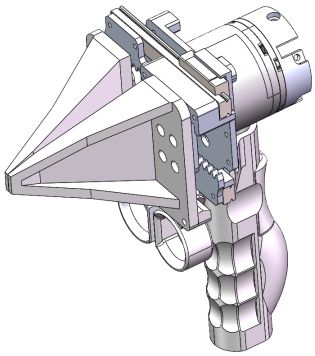
X5&AC one-单臂ROS2话题说明

使用机械臂时，务必确保安装稳定，以基座为轴心一米半径内确保空旷，当心碰撞易碎物品及造成人员损伤。出现紧急情况请先关闭电源。

型号区分

型号		示意图	区别
X5（2023）	标准单臂 launch文件以v1文件夹下		单轨二指夹爪

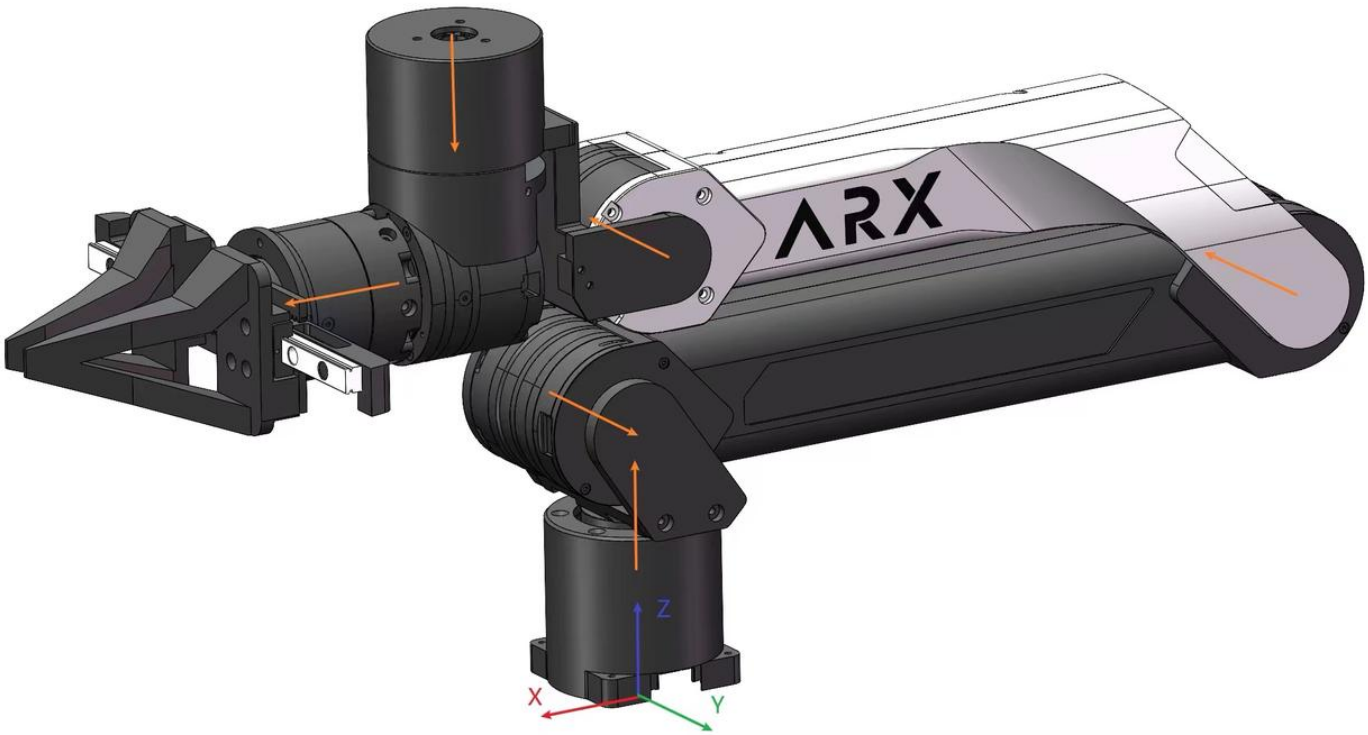
			
X5（2025）	AC one上机械臂 launch文件以v2作为标识		双轨二指夹爪

关节角信息及坐标系

关节范围

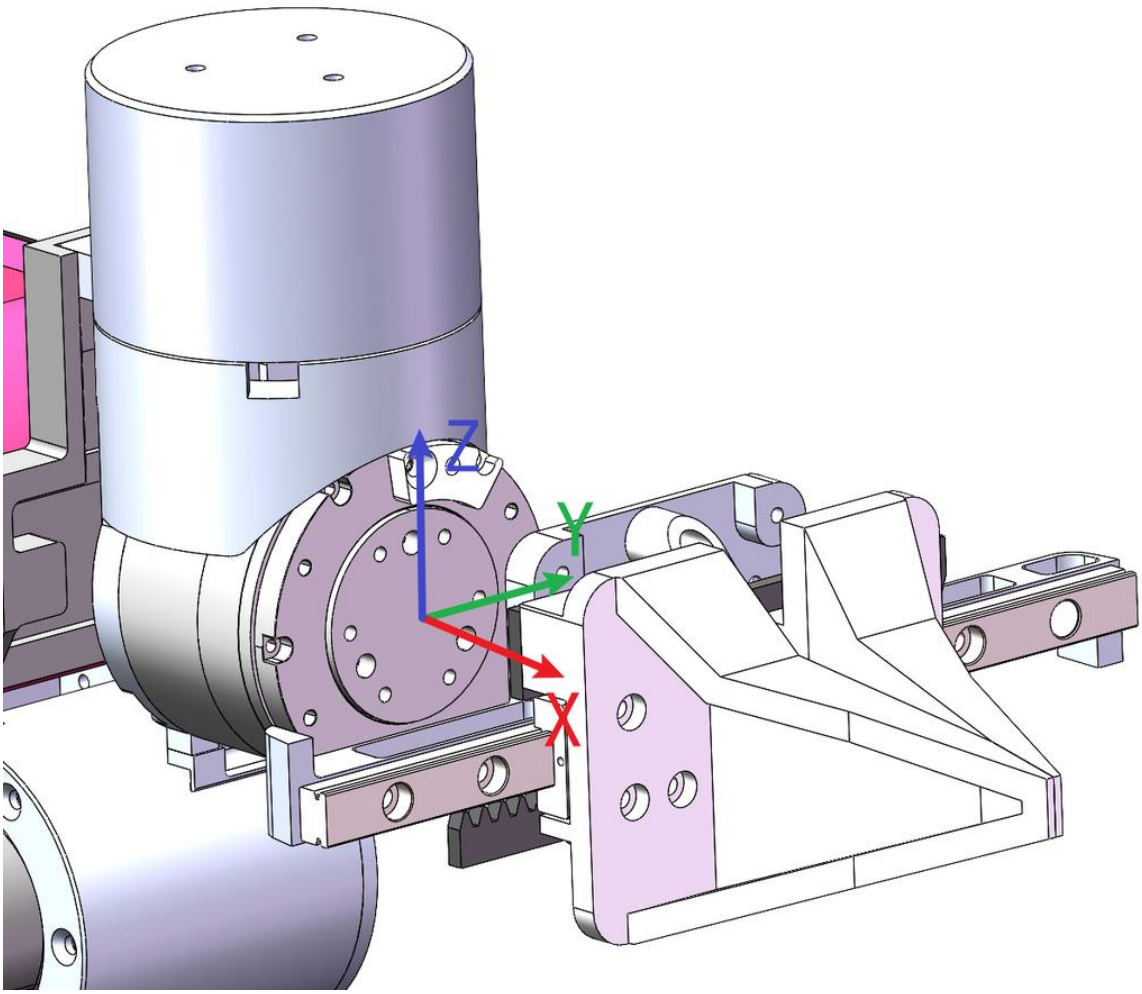
关节	软件限位最小值rad	软件限位最大值rad	机械限位最小值 度	机械限位最大值 度
关节1	-1.57	1.57	-150°	180°
关节2	-0.1（防止解算失效）	3.6	0°	210°
关节3	-0.1（防止解算失效）	3	0°	180°
关节4	-1.29	1.29	-90°	90°
关节5	-1.48	1.48	-90°	90°
关节6	-1.74	1.74	-120°	120°
关节7 (AC one夹爪)	-3.4	0.1（0为闭合，0.1为了夹紧）		
关节7（标准夹爪）	0	5		

关节轴向



关节转向符合右手定理，大拇指的指向关节轴向，四指方向就是电机转动的正方向。

末端坐标系



在初始位置，末端坐标系和参考坐标系重合，位置和姿态都是0，如上图所示。

组装&环境配置

详见组装&环境配置手册

CAN设备启动

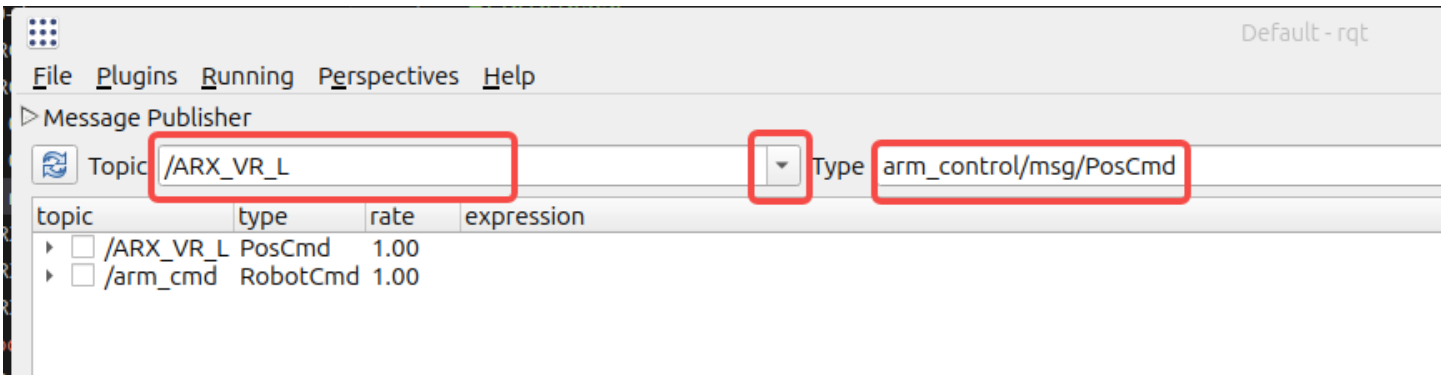
详见ARX-CAN手册

进行以下步骤前，请务必确保掌握以下基础操作

利用rqt进行话题收发

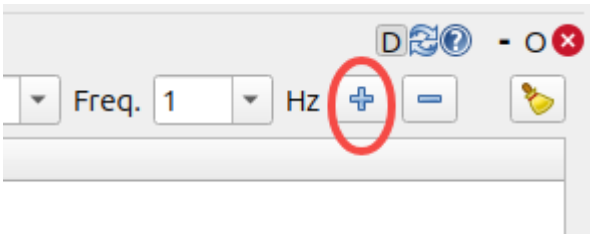
确保在工作空间下，非随意打开终端

```
代码块
1   source install/setup.bash
2   rqt
```

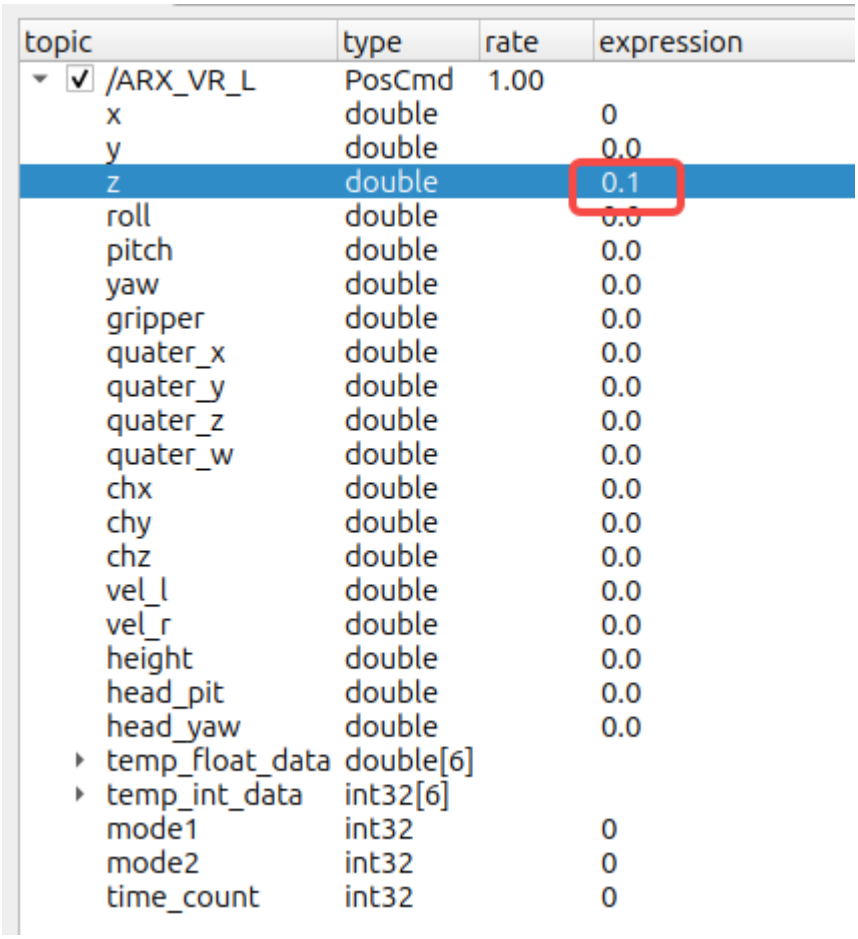


在topic处选择对应话题，确保type类型与其一致。

点击右上角+号，添加对应话题



展开后，在对应位置设定数值即可完成发送命令，注意工作空间，不要超过额定数值



topic	type	rate	expression
/ARX_VR_L	PosCmd	1.00	
x	double		0
y	double		0.0
z	double		0.1
roll	double		0.0
pitch	double		0.0
yaw	double		0.0
gripper	double		0.0
quater_x	double		0.0
quater_y	double		0.0
quater_z	double		0.0
quater_w	double		0.0
chx	double		0.0
chy	double		0.0
chz	double		0.0
vel_l	double		0.0
vel_r	double		0.0
height	double		0.0
head_pit	double		0.0
head_yaw	double		0.0
temp_float_data	double[6]		
temp_int_data	int32[6]		
mode1	int32		0
mode2	int32		0
time_count	int32		0

紧急情况

人员使用时请远离工作空间，避免损伤。若出现紧急情况，请先断电处理。

文件目录

	功能
ARX_CAN	CAN设备配置
ARX_VR_SDK	VR通讯
00-sh	编译及快捷启动脚本

以下操作前务必开启相关CAN

手臂 （注意确保只有一个控制终端运行）

控制

关节控制

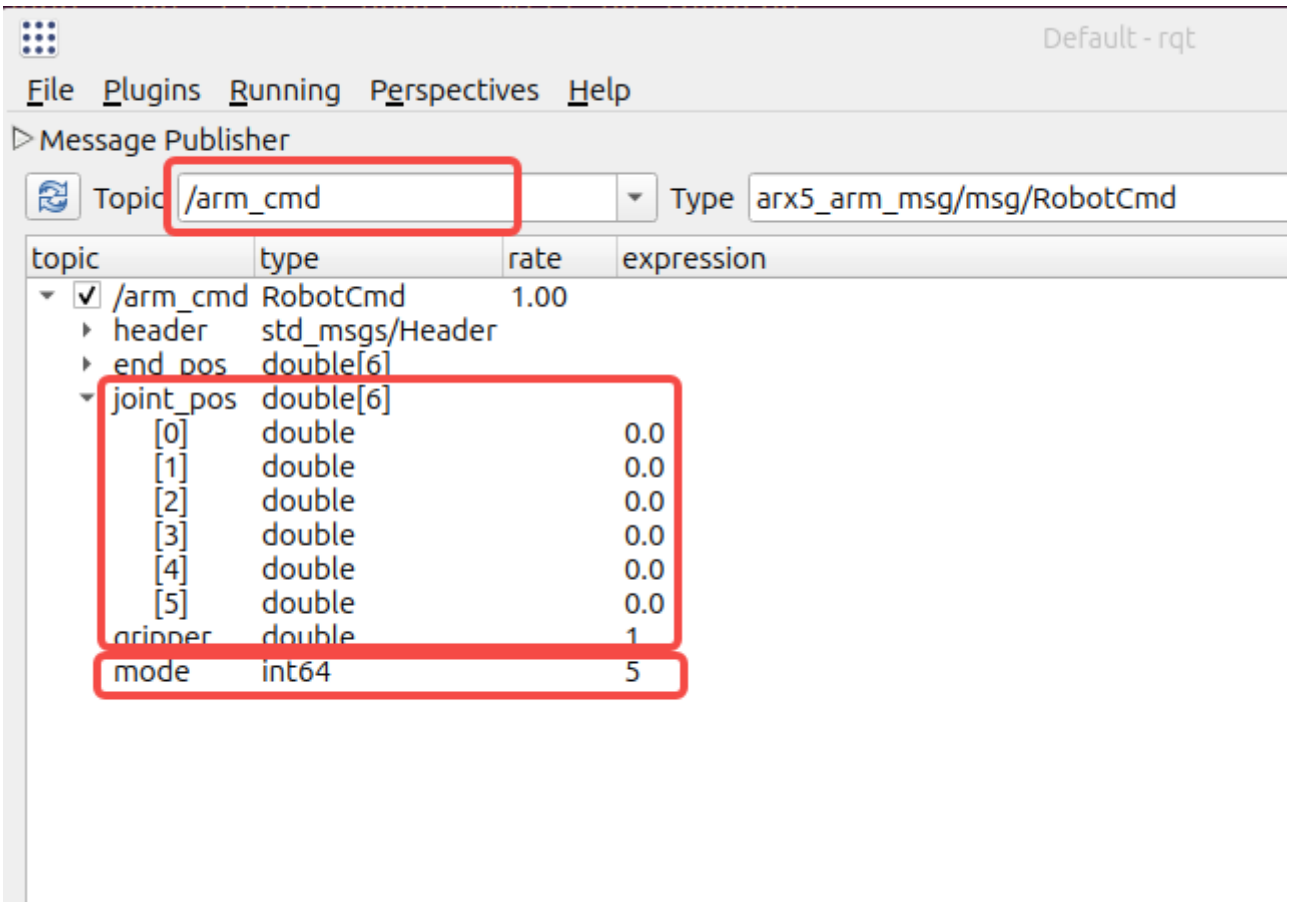
进入ARX_X5/ROS2/X5_ws

```
source install/setup.bash

X5-2023
ros2 launch arx_x5_controller open_single_arm.launch.py

X5-2025
ros2 launch arx_x5_controller v2_single_arm.launch.py
```

进入ARX_X5/ROS2/X5_ws另开终端

```
source install/setup.bash
rqt
```

除夹爪，设置角度建议不要超过+-0.1来验证链路



姿态位置控制

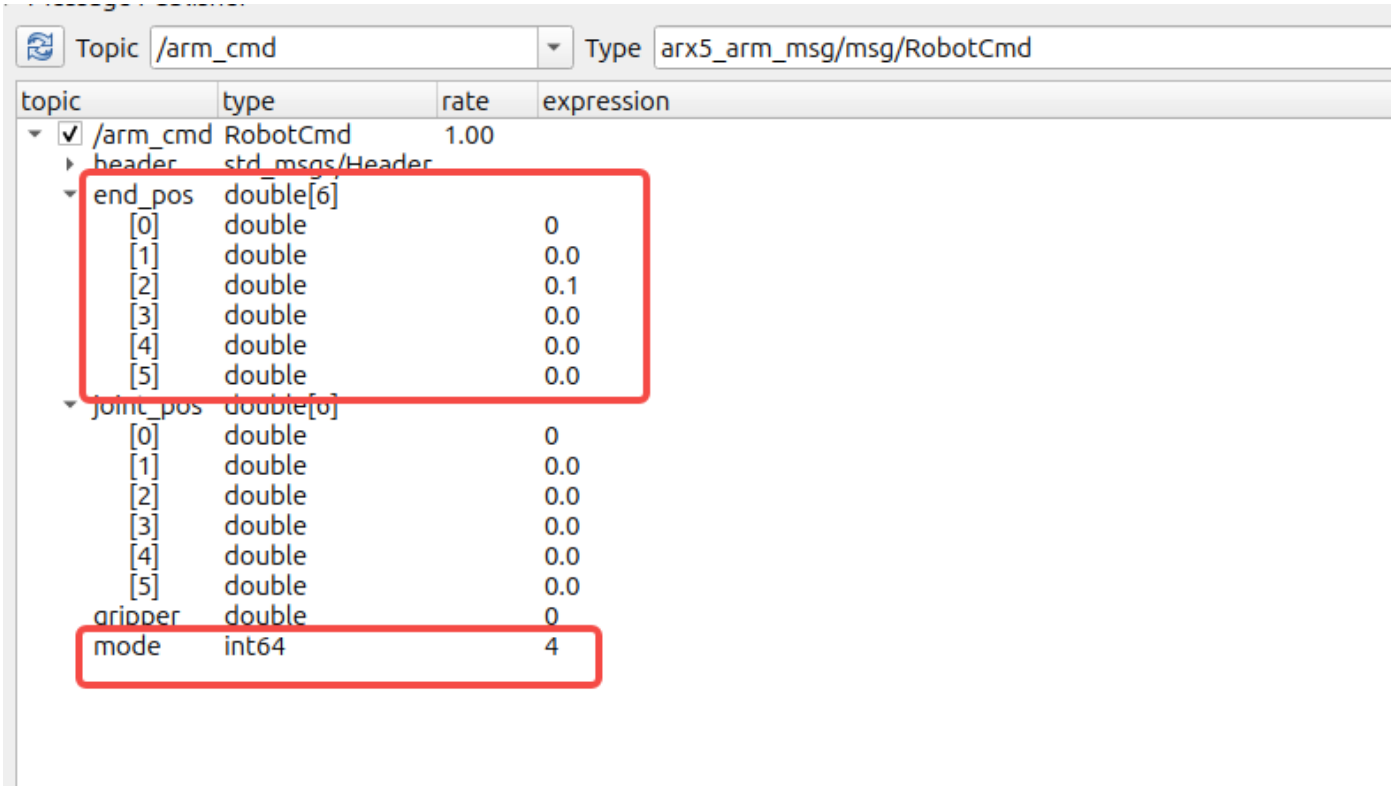
进入ARX_X5/ROS2/X5_ws

```
source install/setup.bash

X5-2023
ros2 launch arx_x5_controller open_single_arm.launch.py

X5-2025
ros2 launch arx_x5_controller v2_single_arm.launch.py
```

进入ARX_X5/ROS2/X5_ws另开终端

```
source install/setup.bash
rqt
```

end_pos中 0-5 对应x y z roll pitch yaw



重力补偿模式

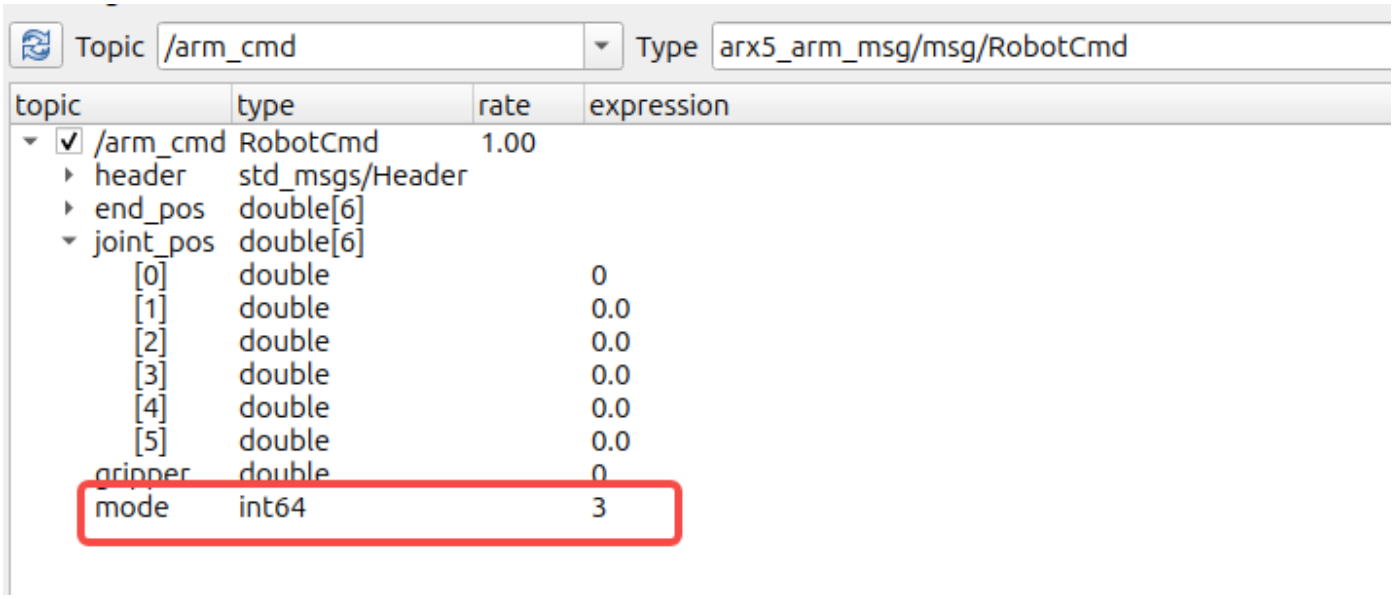
进入ARX_X5/ROS2/X5_ws

```
source install/setup.bash
```

```
2
3   X5-2023
4   ros2 launch arx_x5_controller open_single_arm.launch.py
5
6   X5-2025
7   ros2 launch arx_x5_controller v2_single_arm.launch.py
```

进入ARX_X5/ROS2/X5_ws另开终端

```
代码块
1   source install/setup.bash
2   rqt
```



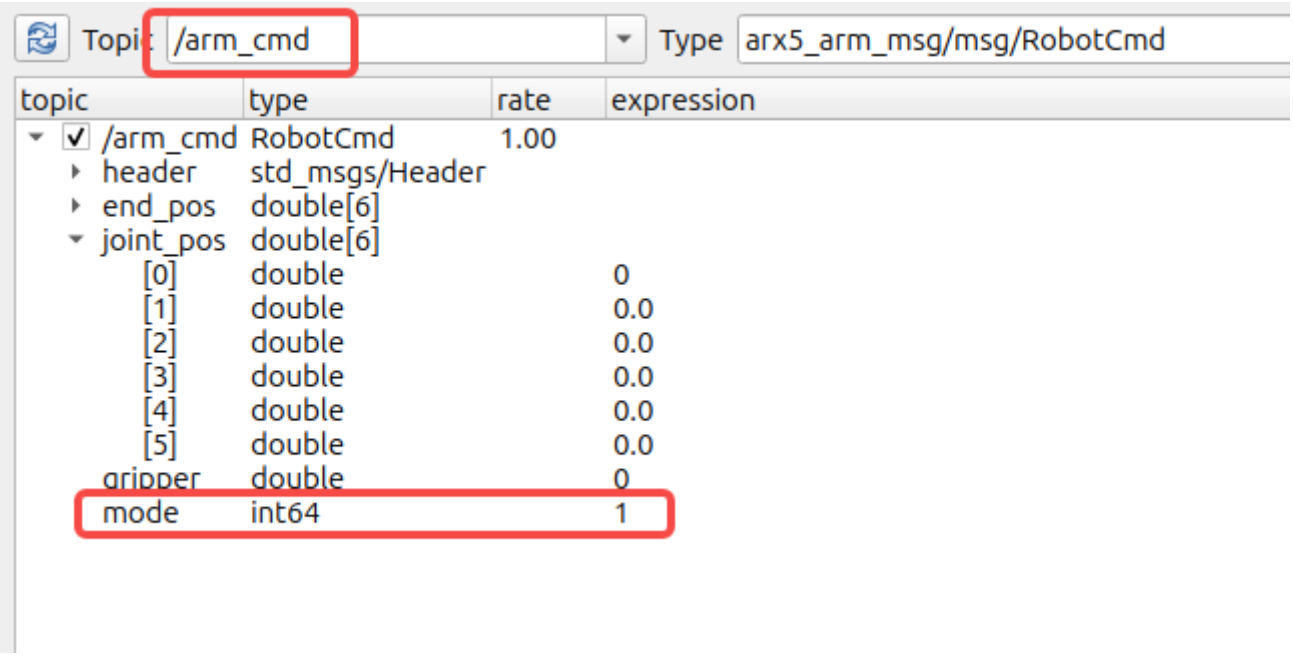
复位

进入ARX_X5/ROS2/X5_ws

```
代码块
1   source install/setup.bash
2
3   X5-2023
4   ros2 launch arx_x5_controller open_single_arm.launch.py
5
6   X5-2025
7   ros2 launch arx_x5_controller v2_single_arm.launch.py
```

进入ARX_X5/ROS2/X5_ws另开终端

```
source install/setup.bash
rqt
```



阻尼模式

进入ARX_X5/ROS2/X5_ws

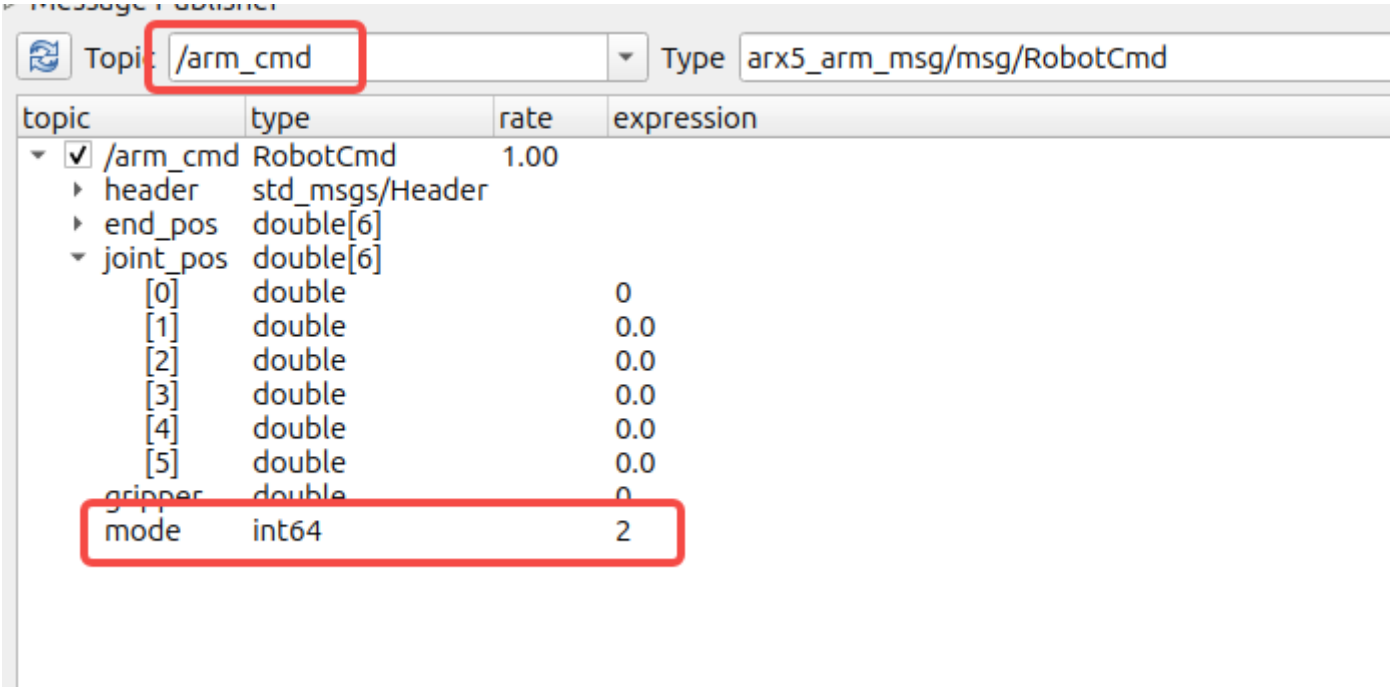
代码块

```
source install/setup.bash

X5-2023
ros2 launch arx_x5_controller open_single_arm.launch.py

X5-2025
ros2 launch arx_x5_controller v2_single_arm.launch.py
```

进入ARX_X5/ROS2/X5_ws另开终端

代码块

```
source install/setup.bash
rqt
```



反馈

关节反馈

在启动对应控制命令后

进入ARX_X5/ROS2/X5_ws另开终端

代码块

```
source install/setup.bash
ros2 topic echo /arm_status
```

```
header:
  stamp:
    sec: 1765363339
    nanosec: 466845457
  frame_id: ''
end_pos:
- 2.486444653154729e-05
- -3.8198192558198 26e-05
- 0.0010755449498 01055
- -0.0074528067100 44608
- 0.0009536783639875621
- 2.828447111991652e-08
joint_pos:
- -0.00057220458984375
- 0.00476837158203125
- 0.00438690185546875
- -0.00057220458984375
- -0.00057220458984375
- -0.00743865966796875
- 0.9018087387084961
joint_vel:
- 0.013187408447265625
- -0.021978378295898438
- -0.0043964385986328125
- -0.010990142822265625
- 0.010990142822265625
- -0.03296661376953125
- 0.010990142822265625
joint_cur:
- -0.0073261260986328125
- 0.0073261260986328125
- 2.7326011657714844
- 1.6532354354858398
- -0.0073261260986328125
- 0.05616569519042969
- -0.021978378295898438
---
```

姿态反馈

在启动对应控制命令后

进入ARX_X5/ROS2/X5_ws另开终端

代码块

```
source install/setup.bash
ros2 topic echo /arm_status
```

```
header:
  stamp:
    sec: 1765364199
    nanosec: 246850686
  frame_id: ''
end_pos:
- -0.00011556782615482208
- -4.81408087244193 65e-05
- 0.100193773276676
- 0.005898452665231465
- 0.0005722059386991231
- -0.0003795528287340 2147
joint_pos:
- -0.00057220458984375
- 0.2645530700683594
- 0.3942546844482422
- -0.13027381896972656
- -0.00019073486328125
- 0.00591278076171875
- -0.0286102294921875
joint_vel:
- -0.0043964385986328125
- -0.013187408447265625
- -0.0043964385986328125
- -0.010990142822265625
- -0.010990142822265625
- -0.010990142822265625
- -0.010990142822265625
joint_cur:
- 0.0073261260986328125
- 0.2417583465576172
- 2.6886444091796875
- 1.7362632751464844
- -0.0073261260986328125
- -0.07570171356201172
- -0.5787544250488281
---
```

mode	模式功能	备注
0	力矩清零	所有关节力矩为0
1	机械臂复位	回到初始位形
2	阻尼模式	在“0”的基础上增加阻尼
3	重力补偿	可任意拖动
4	末端位姿控制	通过“end_pos”控制
5	关节控制	通过“joint_pos”控制

多臂配置（2025版为例）

基础认识

ROS2通过launch文件进行控制，每个launch对应一个yaml文件来进行参数配置。

以X5-2025版为例，launch其目录在
ARX_X5/ROS2/X5_ws/src/arx_x5_ros2/arx_x5_controller/launch下

x5_v2/v2_single_arm.launch.py

ROS2 > X5_ws > src > arx_x5_ros2 > arx_x5_controller > launch > x5_v2 > v2_single_arm.launch.py

```python
import os

from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch_ros.actions import Node

params_file = os.path.join(
    get_package_share_directory('arx_x5_controller'), 'config', 'v2_single_arm.yaml')

arm_node = Node(
    package='arx_x5_controller',
    executable='X5Controller',
    name='arm',
    output='screen',
    parameters=[params_file],
)


def generate_launch_description():
    return LaunchDescription([
        DeclareLaunchArgument(name='params_file',
                              default_value=params_file),
        arm_node,
    ])

```

yaml为ARX_X5/ROS2/X5_ws/src/arx_x5_ros2/arx_x5_controller/config/v2_single_arm.yaml

文件配置

can规则配置

建议左臂can1 右臂can3,详细配置参阅ARX-CAN文档

launch文件更改

复制一份v2_single_arm.launch.py，示例更改为v2_single_arm2.launch.py

更改前内容

ROS2 > X5_ws > src > arx_x5_ros2 > arx_x5_controller > launch > x5_v2 > v2_single_arm.launch.py

```python
import os

from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch_ros.actions import Node

params_file = os.path.join(
    get_package_share_directory('arx_x5_controller'), 'config', 'v2_single_arm.yaml')

arm_node = Node(
    package='arx_x5_controller',
    executable='X5Controller',
    name='arm',
    output='screen',
    parameters=[params_file],
)


def generate_launch_description():
    return LaunchDescription([
        DeclareLaunchArgument(name='params_file',
                              default_value=params_file),
        arm_node,
    ])

```

更改后内容

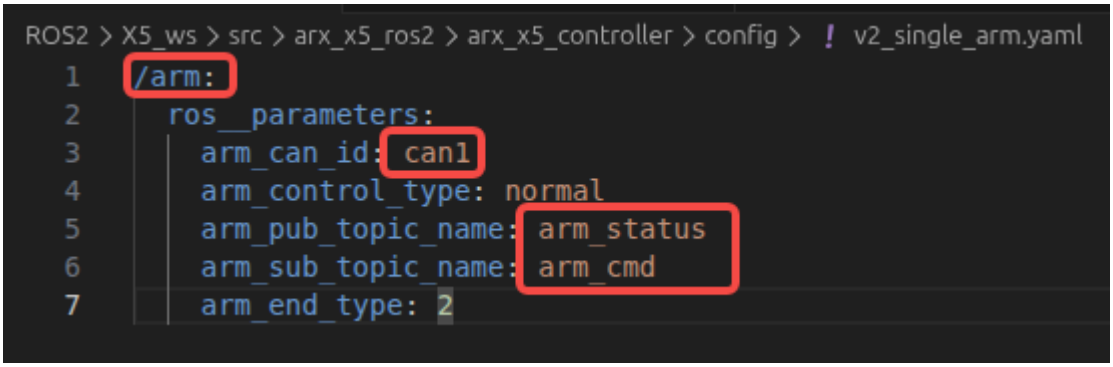
ROS2 > X5_ws > src > arx_x5_ros2 > arx_x5_controller > launch > x5_v2 > v2_single_arm2.launch.py

```python
import os

from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch_ros.actions import Node

params_file = os.path.join(
    get_package_share_directory('arx_x5_controller'), 'config', 'v2_single_arm2.yaml')

arm_node2 = Node(
    package='arx_x5_controller',
    executable='X5Controller',
    name='arm2',
    output='screen',
    parameters=[params_file],
)


def generate_launch_description():
    return LaunchDescription([
        DeclareLaunchArgument(name='params_file',
                              default_value=params_file),
        arm_node2,
    ])

```

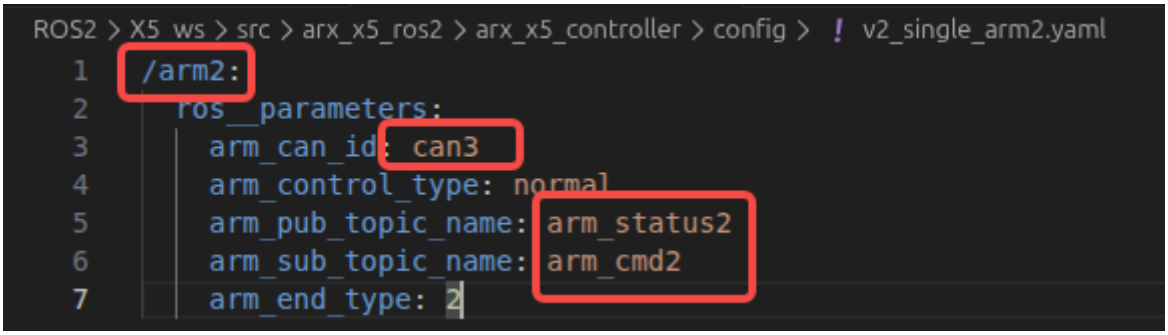
yaml文件更改

复制v2_single_arm.yaml,示例更改为v2_single_arm2.yaml

更改前



更改后



重新编译，启动can设备后，分别运行两个launch即可

异常处理

机械臂垂落，无法控制	终端是否提示safe mode（碰撞检测进入保护模式，断电复位，重启即可）
某个can口打不开	检查can连接，重新插拔对应的usb，重新开启can。
电机无法连接	重新插拔机械臂底座的插头
程序一直在初始化	保证usb接口带宽足够，不要和usb wifi等数据量较大设备公用一个usb
启动机械臂后抖动	确保只有一个控制终端运行，避免多个产生抢占
机械臂非设定运动	确保关闭多机通讯