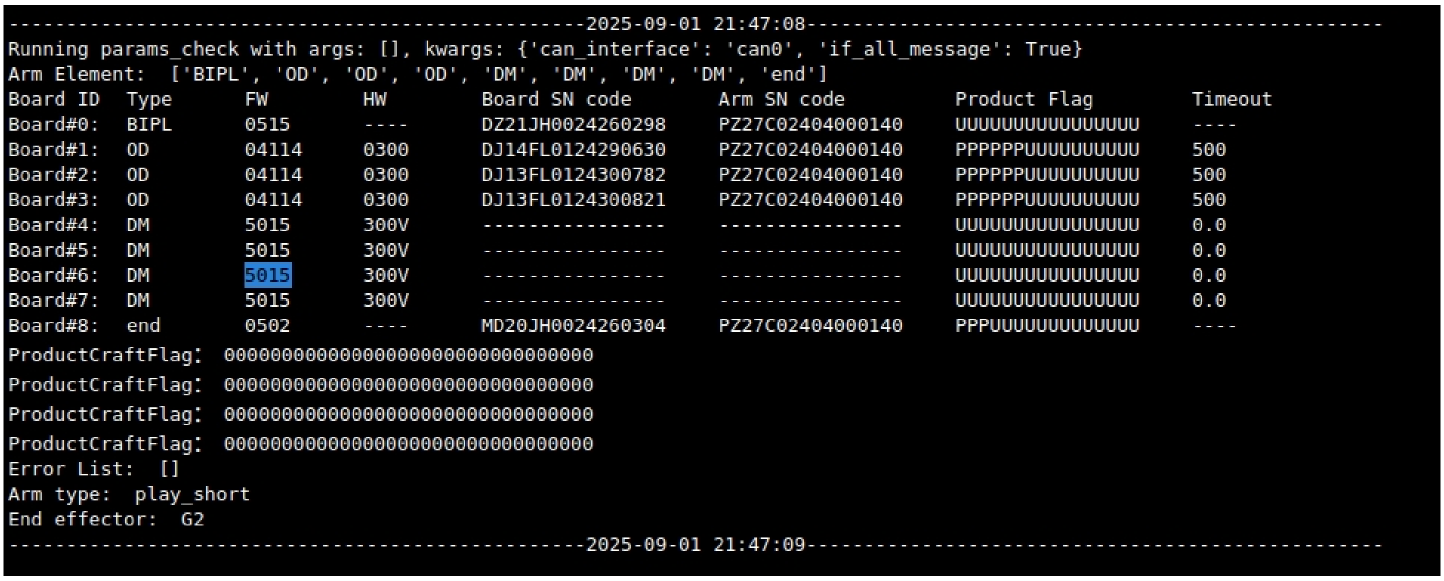
airbot_vr部署操作文档

版本记录

版本	日期	修改记录	编写人	备注
0.0.1	2025.08.29	初版	HanYang	• First version
0.0.2	2025.09.01	操作说明	小昕	

1. 前提

保证Play的固件为最新的版本，版本型号如图：

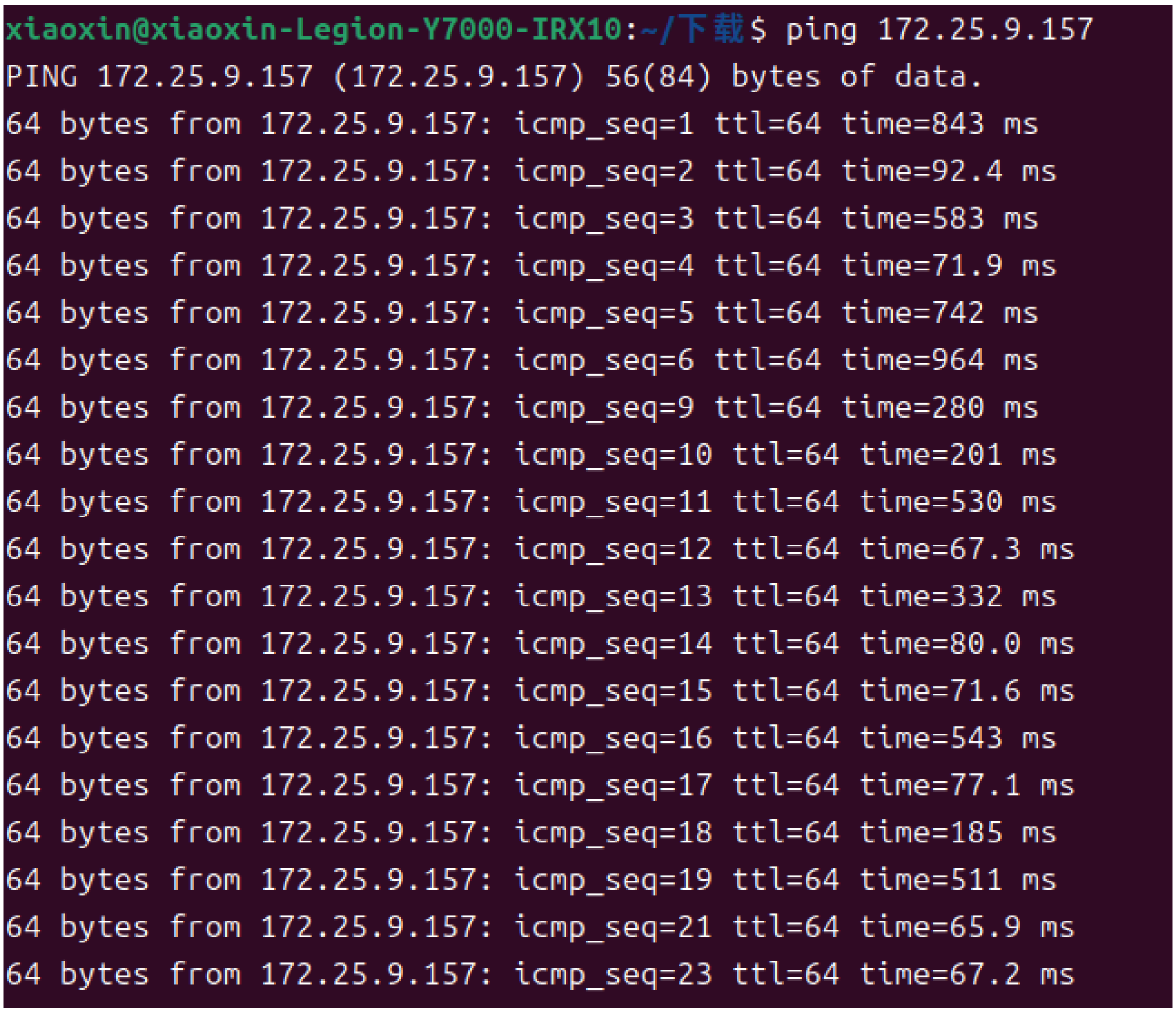


```
------------------------------------------------2025-09-01 21:47:08------------------------------------------------
Running params_check with args: [], kwargs: {'can_interface': 'can0', 'if_all_message': True}
Arm Element:  ['BIPL', 'OD', 'OD', 'OD', 'DM', 'DM', 'DM', 'DM', 'end']
Board ID  Type      FW        HW        Board SN code       Arm SN code         Product Flag        Timeout
Board#0:  BIPL      0515      ----      DZ21JH0024260298    PZ27C02404000140    UUUUUUUUUUUUUUUU    ----
Board#1:  OD        04114     0300      DJ14FL0124290630    PZ27C02404000140    PPPPPPUUUUUUUUUU    500
Board#2:  OD        04114     0300      DJ13FL0124300782    PZ27C02404000140    PPPPPPUUUUUUUUUU    500
Board#3:  OD        04114     0300      DJ13FL0124300821    PZ27C02404000140    PPPPPPUUUUUUUUUU    500
Board#4:  DM        5015      300V      ----------------    ----------------    UUUUUUUUUUUUUUUU    0.0
Board#5:  DM        5015      300V      ----------------    ----------------    UUUUUUUUUUUUUUUU    0.0
Board#6:  DM        5015      300V      ----------------    ----------------    UUUUUUUUUUUUUUUU    0.0
Board#7:  DM        5015      300V      ----------------    ----------------    UUUUUUUUUUUUUUUU    0.0
Board#8:  end       0502      ----      MD20JH0024260304    PZ27C02404000140    PPPUUUUUUUUUUUUU    ----
ProductCraftFlag: 00000000000000000000000000000000
ProductCraftFlag: 00000000000000000000000000000000
ProductCraftFlag: 00000000000000000000000000000000
ProductCraftFlag: 00000000000000000000000000000000
Error List:  []
Arm type:  play_short
End effector:  G2
------------------------------------------------2025-09-01 21:47:09------------------------------------------------
```

网络测试：

ping GO2的无线IP测试延迟

！！！注意：若无线网络延迟出现如下不稳定情况，建议使用单独路由器提供网络或使用有线连接VR设备和GO2



```
xiaoxin@xiaoxin-Legion-Y7000-IRX10:~/下载$ ping 172.25.9.157
PING 172.25.9.157 (172.25.9.157) 56(84) bytes of data.
64 bytes from 172.25.9.157: icmp_seq=1 ttl=64 time=843 ms
64 bytes from 172.25.9.157: icmp_seq=2 ttl=64 time=92.4 ms
64 bytes from 172.25.9.157: icmp_seq=3 ttl=64 time=583 ms
64 bytes from 172.25.9.157: icmp_seq=4 ttl=64 time=71.9 ms
64 bytes from 172.25.9.157: icmp_seq=5 ttl=64 time=742 ms
64 bytes from 172.25.9.157: icmp_seq=6 ttl=64 time=964 ms
64 bytes from 172.25.9.157: icmp_seq=9 ttl=64 time=280 ms
64 bytes from 172.25.9.157: icmp_seq=10 ttl=64 time=201 ms
64 bytes from 172.25.9.157: icmp_seq=11 ttl=64 time=530 ms
64 bytes from 172.25.9.157: icmp_seq=12 ttl=64 time=67.3 ms
64 bytes from 172.25.9.157: icmp_seq=13 ttl=64 time=332 ms
64 bytes from 172.25.9.157: icmp_seq=14 ttl=64 time=80.0 ms
64 bytes from 172.25.9.157: icmp_seq=15 ttl=64 time=71.6 ms
64 bytes from 172.25.9.157: icmp_seq=16 ttl=64 time=543 ms
64 bytes from 172.25.9.157: icmp_seq=17 ttl=64 time=77.1 ms
64 bytes from 172.25.9.157: icmp_seq=18 ttl=64 time=185 ms
64 bytes from 172.25.9.157: icmp_seq=19 ttl=64 time=511 ms
64 bytes from 172.25.9.157: icmp_seq=21 ttl=64 time=65.9 ms
64 bytes from 172.25.9.157: icmp_seq=23 ttl=64 time=67.2 ms
```

连接WIFI：

代码块

```
nmcli device wifi list
sudo nmcli device wifi connect "WiFi名称" password "WiFi密码"
```

2. 安装 AirbotPlay SDK 和 Imgaes

安装臂的镜像，并且启动 Play：

机械臂需要标零

注：第一次会拉取镜像，时间较长

airbot_py-5.1.6-py3-none-any.whl
51.42KB

airbot-configure_5.1.6-1_all.deb
3.38KB

```
airbot_server  -i can0 -p 50051
```

3. 安装airbot_vr_py 软件包

注：提前安装conda

3.1 arm架构在docker内安装以下内容

注:提前配置代理,否则拉取镜像会失败

代码块

```
# docker镜像可通过
docker pull ros:jazzy
# 创建容器
docker run -it   --name robot_arm_vr   --privileged   --network host   -e DISPLAY=$DISPLAY   -e QT_X11_NO_MITSHM=1   -v /tmp/.X11-unix:/tmp/.X11-unix:rw   -v /dev:/dev   -v /etc/localtime:/etc/localtime:ro   -v $HOME/.Xauthority:/root/.Xauthority:ro   -v $HOME/ros_ws:/root/ros_ws   ros:jazzy   /bin/bash
# 写入环境变量
 source /opt/ros/jazzy/setup.bash
# 下载适用于 ARM64 架构的 Miniconda
apt update
apt install wget
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-aarch64.sh -O miniconda.sh
bash miniconda.sh
```

3.2 依赖于 ros2 ，需要安装 py3.12

代码块

```
# 设置清华镜像
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge/
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/msys2/
conda config --set show_channel_urls yes
# 创建conda
```

```
conda create -n airbotplay_312 python=3.12
conda activate airbotplay_312
```

测试

代码块

```
python3 -c 'import rclpy; print("rclpy imported successfully!")'
```

报错 GLIBCXX_3.4.30 error 处理方法，一般是由于conda 环境 gcc相关版本不匹配导致

代码块

```
conda install -c conda-forge libstdcxx-ng
```

创建工作空间：

代码块

```
mkdir -p airbot_vr_py/src/packages
```

在 conda 环境中安装以下包：

此包放在：/home/name/airbot_vr_py/src/packages

airbot_py-5.1.6-py3-none-any.whl
51.42KB

mmk2_kdl_py-0.1.3-py3-none-any.whl
22.56KB

docker需要执行(不使用跳过)

代码块

```
docker cp airbot_py-5.1.6-py3-none-any.whl robot_arm_jazzy:/airbot_vr_py/src/packages
docker cp mmk2_kdl_py-0.1.3-py3-none-any.whl robot_arm_jazzy:/airbot_vr_py/src/packages
```

代码块

```
# 进入对应的conda环境
conda activate airbotplay_312
pip install airbot_py-5.1.6-py3-none-any.whl -i
https://pypi.tuna.tsinghua.edu.cn/simple
pip install mmk2_kdl_py-0.1.3-py3-none-any.whl -i
https://pypi.tuna.tsinghua.edu.cn/simple
```

将此包下载到本地

此包放在: /home/name/airbot_vr_py

airbot_vr_py-0.0.1-py3-none-any.whl
14.35KB

docker使用(否则跳过)

代码块

```
docker cp airbot_vr_py-0.0.1-py3-none-any.whl robot_arm_jazzy:/airbot_vr_py/
```

安装命令:

代码块

```
sudo apt update

# 进入conda环境
# 安装示例
pip install airbot_vr_py-0.0.1-py3-none-any.whl  -i
https://mirrors.huaweicloud.com/repository/pypi/simple
# 安装其他依赖
pip install scipy numpy matplotlib -i
https://mirrors.huaweicloud.com/repository/pypi/simple
```

3.3 运行

代码块

```
python3 -m airbot_vr.vr_arm
```

4. 安装 ROS2-TCP

4.1 部署

代码块

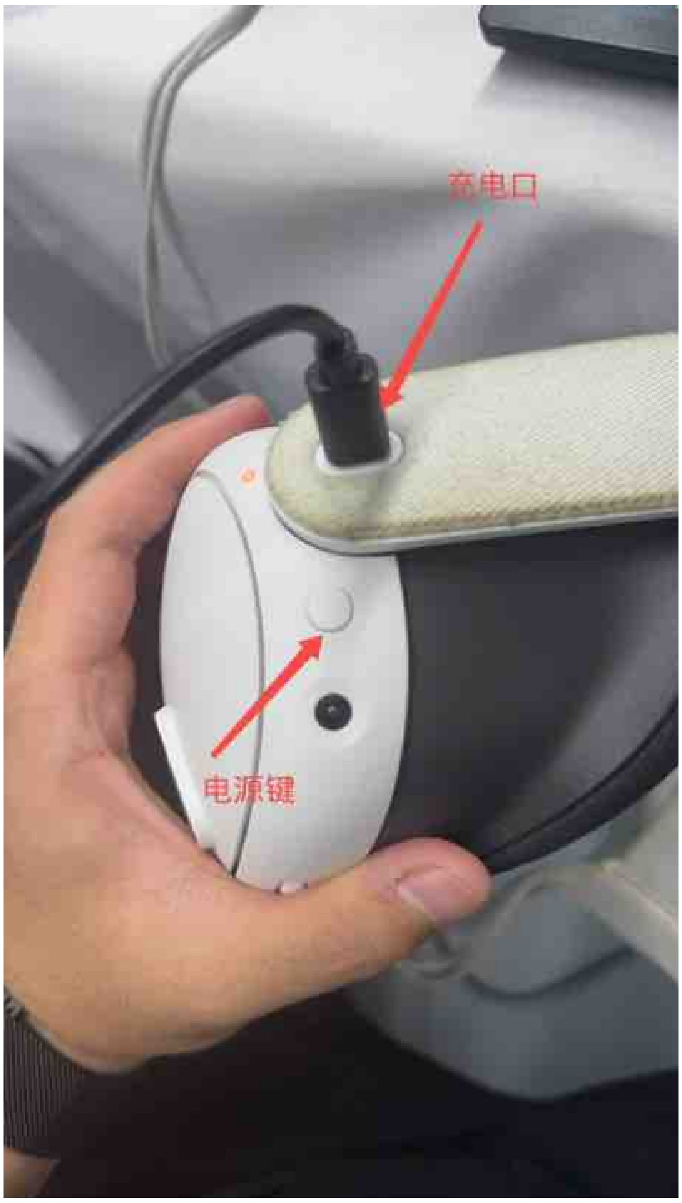
```
# 安装
cd airbot_vr_py/src/
git clone  https://github.com/Unity-Technologies/ROS-TCP-Endpoint.git
cd ROS-TCP-Endpoint
git checkout  main-ros2

# 编译
cd ../..
colcon build --packages-select  ros_tcp_endpoint
source ./install/setup.bash
```

4.2 运行

代码块

```
# 获取本机IP
ifconfig
# 运行ros2 tcp
source /opt/ros/jazzy/setup.bash
source ./install/setup.bash
ros2 run ros_tcp_endpoint default_server_endpoint --ros-args -p ROS_IP:=<your
IP address>
```

4.3 运行Quest3APP

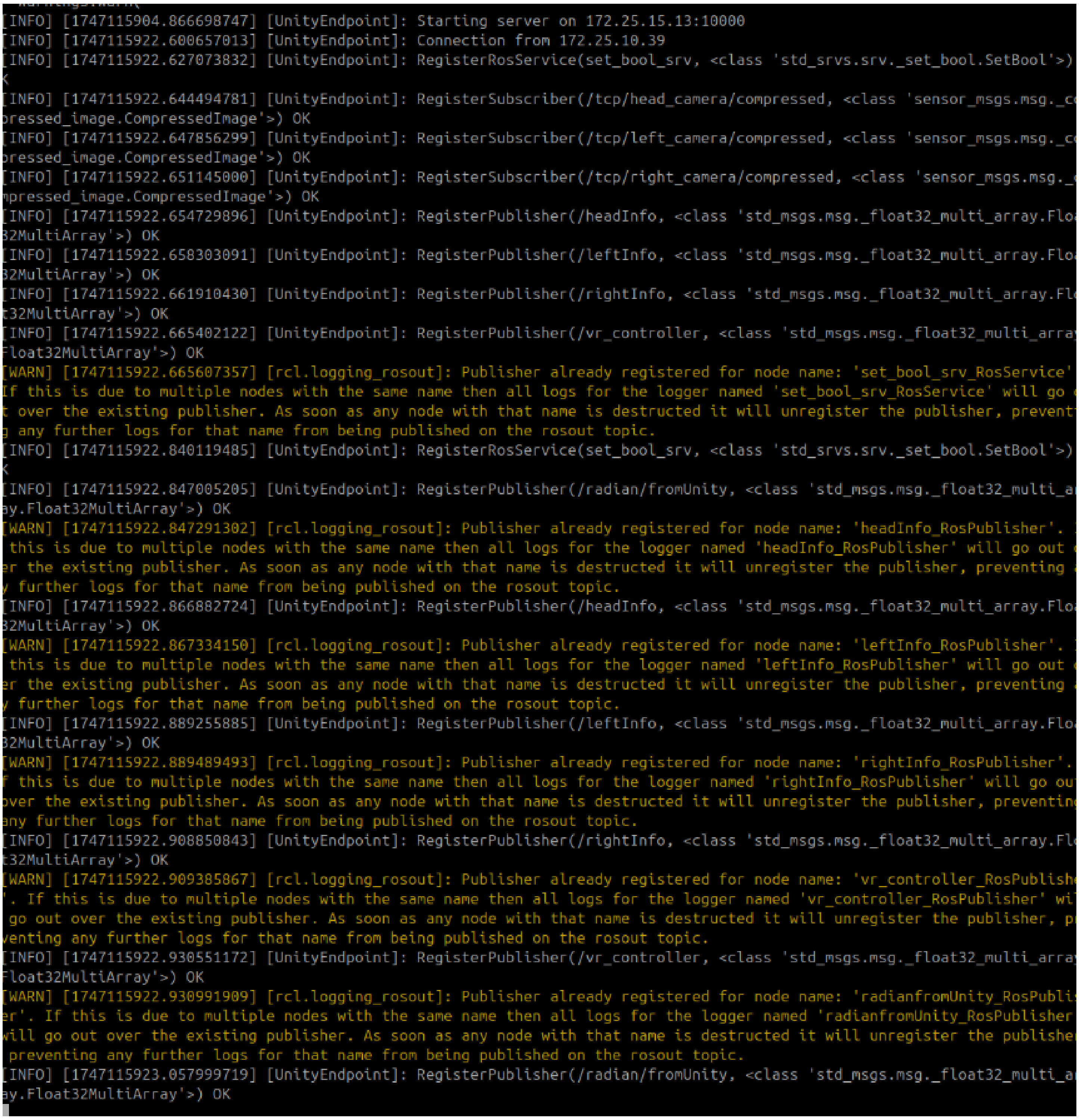


注意！！！：保证机械臂朝向和人体朝向完全一致，否则对应关系会出现问题

开启 VR 设备（长按电源建 2 秒开机，长按 5 秒关机），并且连接 wifi，确保 VR 和 PC 处于同一网络中。

运行VR 设备中的 “VRRobot”APP， 在 IP 配置界面输入上面启动的 ROS_IP 地址，然后点击 Link

#若电脑上运行ros_tcp_endpoint 的终端中出现如下图所示的输出，即为链接成功。



```
  warnings.warn(
[INFO] [1747115904.866698747] [UnityEndpoint]: Starting server on 172.25.15.13:10000
[INFO] [1747115922.600657013] [UnityEndpoint]: Connection from 172.25.10.39
[INFO] [1747115922.627073832] [UnityEndpoint]: RegisterRosService(set_bool_srv, <class 'std_srvs.srv._set_bool.SetBool'>)
K
[INFO] [1747115922.644494781] [UnityEndpoint]: RegisterSubscriber(/tcp/head_camera/compressed, <class 'sensor_msgs.msg._c
pressed_image.CompressedImage'>) OK
[INFO] [1747115922.647856299] [UnityEndpoint]: RegisterSubscriber(/tcp/left_camera/compressed, <class 'sensor_msgs.msg._c
pressed_image.CompressedImage'>) OK
[INFO] [1747115922.651145000] [UnityEndpoint]: RegisterSubscriber(/tcp/right_camera/compressed, <class 'sensor_msgs.msg._
mpressed_image.CompressedImage'>) OK
[INFO] [1747115922.654729896] [UnityEndpoint]: RegisterPublisher(/headInfo, <class 'std_msgs.msg._float32_multi_array.Flo
32MultiArray'>) OK
[INFO] [1747115922.658303091] [UnityEndpoint]: RegisterPublisher(/leftInfo, <class 'std_msgs.msg._float32_multi_array.Flo
32MultiArray'>) OK
[INFO] [1747115922.661910430] [UnityEndpoint]: RegisterPublisher(/rightInfo, <class 'std_msgs.msg._float32_multi_array.Fl
t32MultiArray'>) OK
[INFO] [1747115922.665402122] [UnityEndpoint]: RegisterPublisher(/vr_controller, <class 'std_msgs.msg._float32_multi_array
Float32MultiArray'>) OK
[WARN] [1747115922.665607357] [rcl.logging_rosout]: Publisher already registered for node name: 'set_bool_srv_RosService'
If this is due to multiple nodes with the same name then all logs for the logger named 'set_bool_srv_RosService' will go 
t over the existing publisher. As soon as any node with that name is destructed it will unregister the publisher, prevent
g any further logs for that name from being published on the rosout topic.
[INFO] [1747115922.840119485] [UnityEndpoint]: RegisterRosService(set_bool_srv, <class 'std_srvs.srv._set_bool.SetBool'>)
K
[INFO] [1747115922.847005205] [UnityEndpoint]: RegisterPublisher(/radian/fromUnity, <class 'std_msgs.msg._float32_multi_a
ay.Float32MultiArray'>) OK
[WARN] [1747115922.847291302] [rcl.logging_rosout]: Publisher already registered for node name: 'headInfo_RosPublisher'. 
 this is due to multiple nodes with the same name then all logs for the logger named 'headInfo_RosPublisher' will go out 
er the existing publisher. As soon as any node with that name is destructed it will unregister the publisher, preventing 
y further logs for that name from being published on the rosout topic.
[INFO] [1747115922.866882724] [UnityEndpoint]: RegisterPublisher(/headInfo, <class 'std_msgs.msg._float32_multi_array.Flo
32MultiArray'>) OK
[WARN] [1747115922.867334150] [rcl.logging_rosout]: Publisher already registered for node name: 'leftInfo_RosPublisher'. 
 this is due to multiple nodes with the same name then all logs for the logger named 'leftInfo_RosPublisher' will go out 
er the existing publisher. As soon as any node with that name is destructed it will unregister the publisher, preventing 
y further logs for that name from being published on the rosout topic.
[INFO] [1747115922.889255885] [UnityEndpoint]: RegisterPublisher(/leftInfo, <class 'std_msgs.msg._float32_multi_array.Flo
32MultiArray'>) OK
[WARN] [1747115922.889489493] [rcl.logging_rosout]: Publisher already registered for node name: 'rightInfo_RosPublisher'.
f this is due to multiple nodes with the same name then all logs for the logger named 'rightInfo_RosPublisher' will go ou
over the existing publisher. As soon as any node with that name is destructed it will unregister the publisher, preventin
any further logs for that name from being published on the rosout topic.
[INFO] [1747115922.908850843] [UnityEndpoint]: RegisterPublisher(/rightInfo, <class 'std_msgs.msg._float32_multi_array.Fl
t32MultiArray'>) OK
[WARN] [1747115922.909385867] [rcl.logging_rosout]: Publisher already registered for node name: 'vr_controller_RosPublishe
'. If this is due to multiple nodes with the same name then all logs for the logger named 'vr_controller_RosPublisher' wi
 go out over the existing publisher. As soon as any node with that name is destructed it will unregister the publisher, p
venting any further logs for that name from being published on the rosout topic.
[INFO] [1747115922.930551172] [UnityEndpoint]: RegisterPublisher(/vr_controller, <class 'std_msgs.msg._float32_multi_array
Float32MultiArray'>) OK
[WARN] [1747115922.930991909] [rcl.logging_rosout]: Publisher already registered for node name: 'radianfromUnity_RosPubli
er'. If this is due to multiple nodes with the same name then all logs for the logger named 'radianfromUnity_RosPublisher
will go out over the existing publisher. As soon as any node with that name is destructed it will unregister the publishe
 preventing any further logs for that name from being published on the rosout topic.
[INFO] [1747115923.057999719] [UnityEndpoint]: RegisterPublisher(/radian/fromUnity, <class 'std_msgs.msg._float32_multi_a
ay.Float32MultiArray'>) OK
```

测试按键响应，按下表按键依次测试是否有响应，有响应则通信成功：

代码块

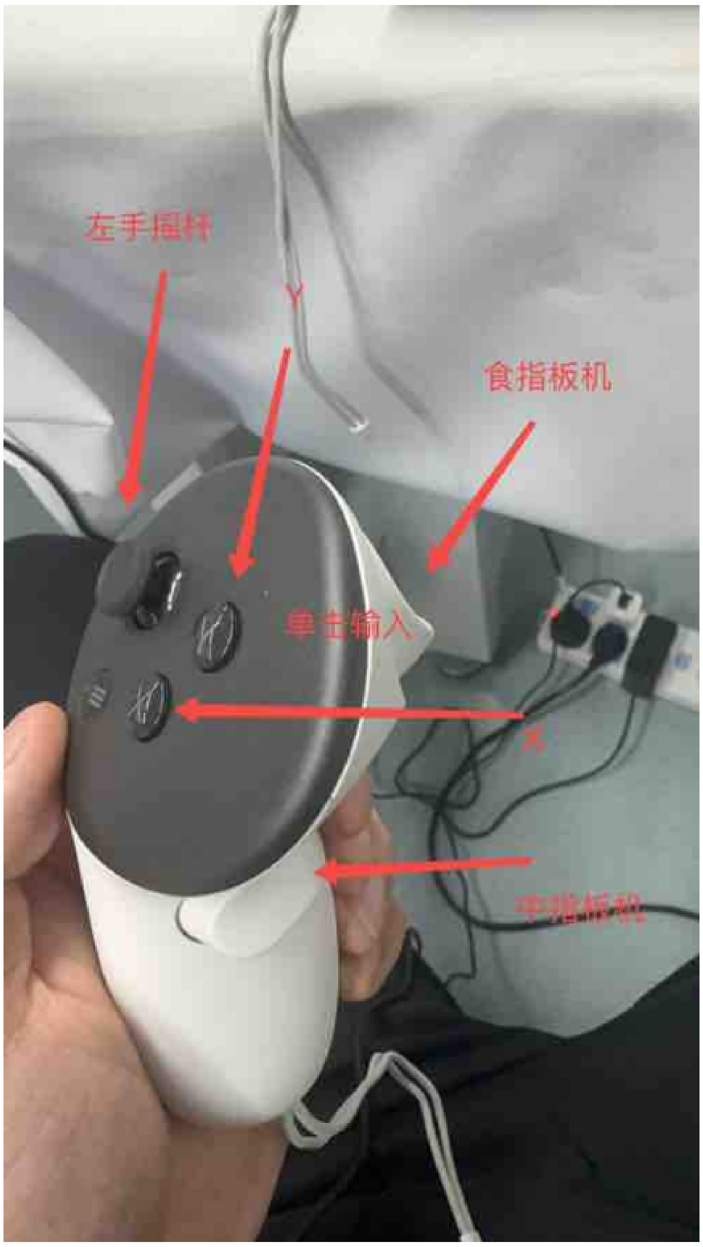
```
# (new terminal)  右下窗口
ros2 topic echo /vr_controller
```

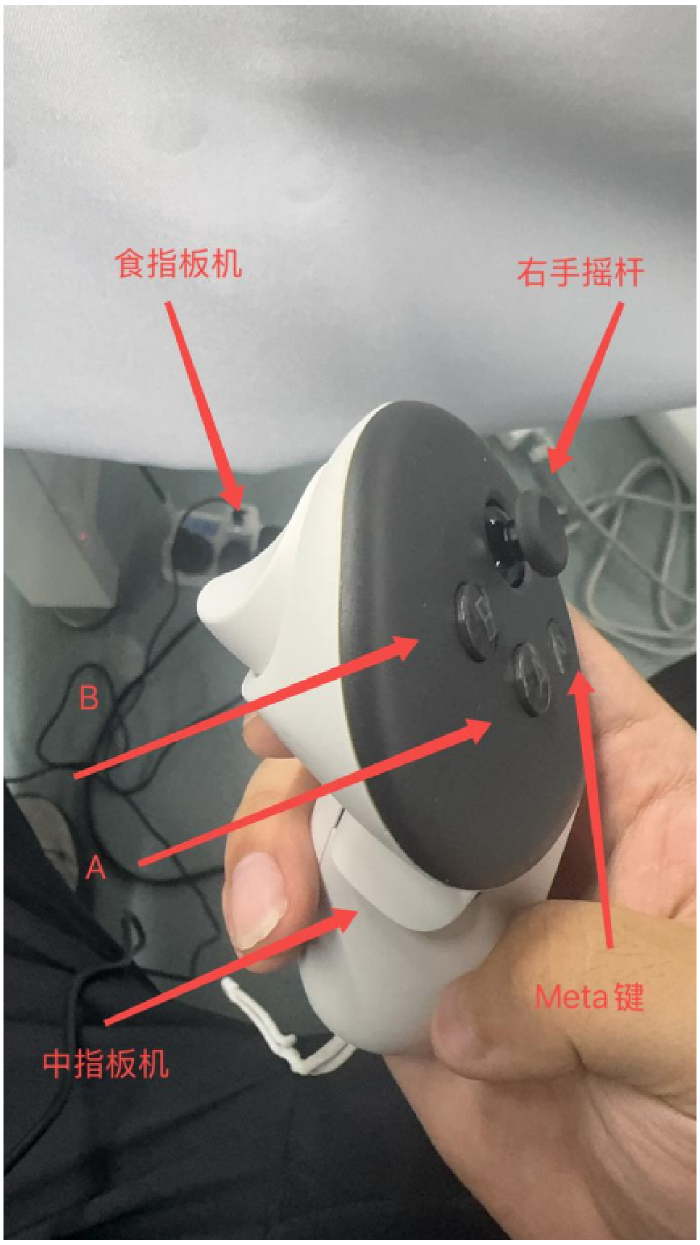
位置	按键	取值（初始 ---> 触发）
0	右手柄 A 键	0 ---> 1

1	右手柄 B 键	0 ---> 1
2	左手柄摇杆（上，下）	-1 ---> 0 ---->1
3	左手柄摇杆（左，右）	-1 ---> 0 ---->1
4	右手柄摇杆（上，下）	-1 ---> 0 ---->1
5	右手柄摇杆（左，右）	-1 ---> 0 ---->1
6	左手柄前扳机	0 ---> 1
7	右手柄前扳机	0 ---> 1
8	左手柄侧方中指扳机	0 ---> 1
9	右手柄侧方中指扳机	0 ---> 1
10	左手柄 X 键	0 ---> 1
11	左手柄 Y 键	0 ---> 1

5. 操作说明

- 长按右手柄的“Meta”按键 3 秒进行复位，然后按下右手柄的“A”键开始同步，此时终端应打印“startVRcontrol”信息。
- 然后按下左手柄食指扳机，此时机械臂末端位置将会跟随左手柄位置进行移动。（期间需要保持左手柄食指扳机一直处于被按下的状态，若松开食指扳机，则机械臂末端将不会跟随左手柄位置移动。）
- 夹爪可通过左手柄中指板机控制开合
- 左手柄上的“X”按键为复位键，按下后机械臂自动回到零位。
- 右手手柄“B”键为取消同步按键。





6. 注意事项

1. 保证人体朝向和机械臂朝向一致！！！
2. 每次使用头显时需确保连接网络后再打开 APP，头显在一段时间不使用会自动断开 wifi 连接
3. 头显在取下后会自动息屏，需要对头显内部的摄像头进行遮挡。
4. 头显在一段时间不使用会自动息屏，需要按下左侧电源键进行唤醒。
5. 每次使用前最好长按 meta 键进行复位。
6. 放下手柄前需按 B 键取消同步。
7. 运行时暂停和启动同步时可能存在位置偏置 bug，推荐在松开中指扳机后间隔两秒左右在按下中指扳机开始同步。
8. 若机械臂上电黄灯闪烁则需要标零。
9. 若出现代码无法控制机械臂的情况，推荐机械臂重新上电再重启节点。

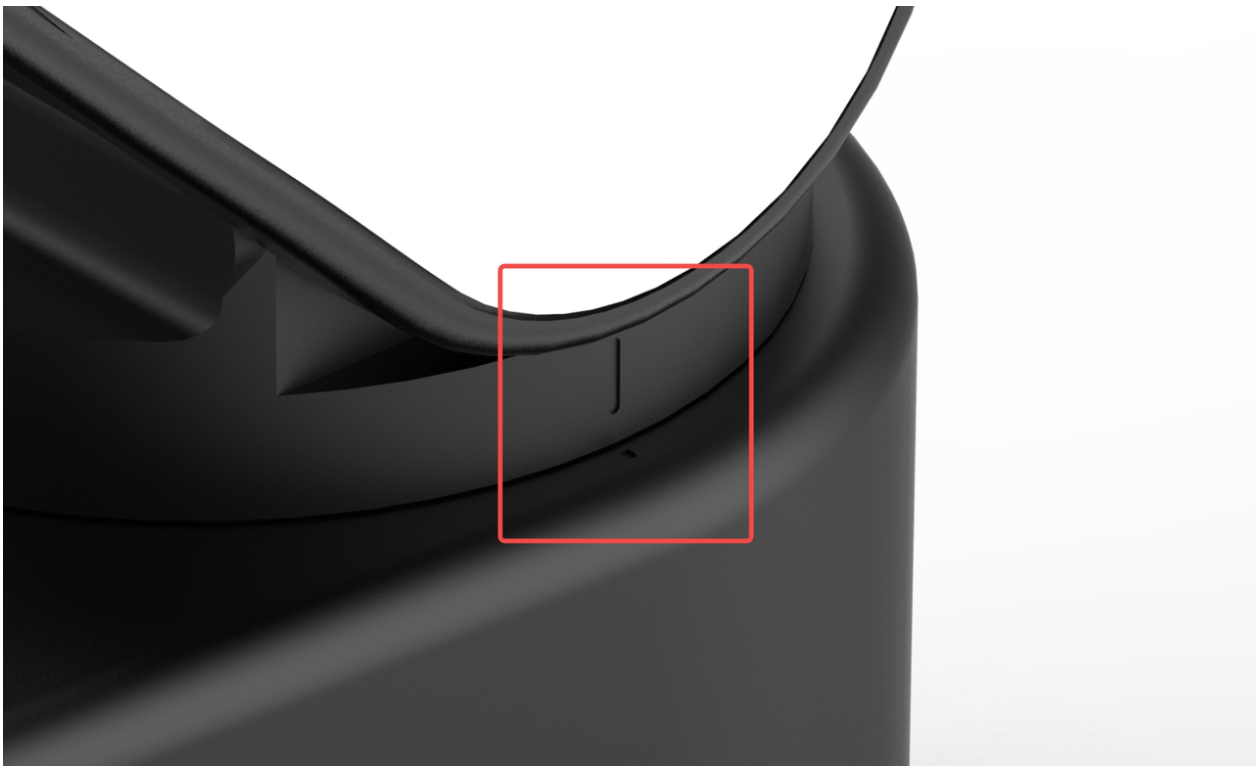
10. 若在使用机械臂过程中发现机械臂失控，则立即松开左手食指扳机并按 X 键复位。
11. 在开始遥操前，先确定自己身前位置大小是否合适，任务中最好不要出现松开左右食指扳机然后再按下的情况。若出现位置无法判断的问题，果断复位。

7. 常见问题

1. 机械臂丢零：

AIRBOT Play，如果上电稳定后，LED 状态灯条呈黄色呼吸状态（黄色闪烁），表示 AIRBOT 检测到零点参考丢失，需重新校零：

- 长按底座按键3秒钟，听到类似“咯哒”声，电机进入自由驱动状态，且关节摩擦力较大；
- 手动将机械臂拖动到零点位置（参考机械臂上的标零刻度上下对齐，4,5,6号关节可随意放置）；



- 再次按下底座按键，再次听到类似“咯哒”声，完成标定。

标定完成后，状态灯条会呈现白色呼吸状态（如果连接了USB线），或者呈白色常亮状态（如果未连接USB线）。