RB-A630-2 同构遥操使用说明

目录

- 1. 软件简介
- 2. 文档目标与适用范围
- 3. 高风险提示
  - 3.1 启动前必查
  - 3.2 运行中风险
- 4. 功能拓扑
  - 4.1 网页快速上手
- 5. 命令行启动与检查
  - 5.1 环境准备
  - 5.2 启动一对机械臂
  - 5.3 启动四个机械臂
  - 5.4 停止流程
- 6. 运行确认与成功判据
- 7. 常见问题与异常处理
- 8. 相关路径

1. 软件简介

RB-A630-2 同构遥操用于四个六轴机械臂之间的主从关节空间跟随。系统中共有 robot_1 、 robot_2 、 robot_3 、 robot_4 四个机械臂实例，其中 robot_1 和 robot_2 作为主臂进行零力拖动， robot_3 和 robot_4 分别作为从臂跟随对应主臂。

当前固定配对为：

- 配对 1： robot_1 主臂零力拖动， robot_3 从臂跟随。
- 配对 2： robot_2 主臂零力拖动， robot_4 从臂跟随。

同构遥操节点负责读取主臂关节角，并向从臂 position 控制器发送目标位置。主臂零力拖动力矩由 raybot_dynamics_control 中的 RB-A630-2 零力拖动节点输出。

2. 文档目标与适用范围

本文档用于指导现场调试与使用人员完成 RB-A630-2 同构遥操的启动、运行确认、停止和异常处理。

适用范围：

- 工作区： /home/raybot/raybot_core_ws
- 模型工作区： /home/raybot/raybot_models
- 同构遥操软件包： raybot_homogeneous_teleop
- 零力拖动软件包： raybot_dynamics_control
- 机器人型号： rb_a630_2
- 机械臂实例： robot_1 、 robot_2 、 robot_3 、 robot_4

不适用范围：

- 同构遥操算法设计与二次开发说明
- 零力拖动参数标定细节
- 非 RB-A630-2 四机械臂拓扑

3. 高风险提示

> ***高风险提示***：*同构遥操会让从臂跟随主臂运动，具备真实机械运动风险。启动前必须确认四个机械臂运动范围内无人停留，急停按钮可用，并默认软件链路可能出现延迟、方向错误、状态丢失或异常退出。出现误动作、碰撞风险、人员风险或状态反馈异常时，物理急停优先于任何命令行操作。*

3.1 启动前必查

启动前必须确认：

- 四个机械臂危险运动区域已清空。
- 主臂零力拖动启动前有人托住机械臂，避免主臂因重力或参数不匹配快速下落。
- 物理急停可立即触达。
- ROS2 Humble 环境、 raybot_models 和 raybot_core_ws 均已正确加载。
- 当前系统中不存在旧的 /controller_manager 。
- /joint_states 能持续刷新。
- /raybot_robot/hardware_ready 能变为 true 。
- 首次运行必须先单对、小幅度、低风险方向验证。

如果 Ray Studio 或其他终端已经启动机器人控制，需要先停止旧控制链路，避免重复启动 /controller_manager 。

可使用：

```
curl -fsS http://127.0.0.1:8889/api/robot-control/stop || true
```

3.2 运行中风险

运行过程中尤其注意：

- robot_3 会跟随 robot_1 ， robot_4 会跟随 robot_2 ，从臂附近不得站人。
- 主臂零力拖动不等于完全无重力、完全悬停或绝对安全。
- 从臂跟随存在控制周期、通信、状态刷新和限速带来的延迟。
- follow_mode:=absolute 下，主从初始关节角差异过大时，遥操节点会拒绝启动或可能带来较大跟随目标变化。
- 未确认主从方向和配对关系前，不得进行大幅拖动。
- 任一主臂零力拖动异常、从臂抖动、跟随方向异常或状态超时，应立即停机。

4. 功能拓扑

控制关系如下：

角色	机械臂	控制方式	控制器
主臂 1	robot_1	零力拖动	robot_1_arm_eff_controller
主臂 2	robot_2	零力拖动	robot_2_arm_eff_controller
从臂 1	robot_3	位置跟随	robot_3_arm_pos_controller
从臂 2	robot_4	位置跟随	robot_4_arm_pos_controller

关键节点与话题：

项目	名称
单对遥操 launch	single_pair_homogeneous_teleop.launch.py
四臂遥操 launch	four_arms_homogeneous_teleop.launch.py
零力拖动 launch	zero_force_drag_rb_a630_2.launch.py
主臂 1 零力节点	zero_force_drag_robot_1
主臂 2 零力节点	zero_force_drag_robot_2
主臂 1 effort 命令	/robot_1_arm_eff_controller/commands
主臂 2 effort 命令	/robot_2_arm_eff_controller/commands
从臂 1 position 命令	/robot_3_arm_pos_controller/commands
从臂 2 position 命令	/robot_4_arm_pos_controller/commands

4.1 网页快速上手

如果现场已经接入 Ray Studio，推荐从 机器人控制 页的 拖动示教 区域进入同构遥操面板。

参考下图所示顺序操作：



推荐步骤如下：

1. 打开 机器人控制 页，点击 拖动示教 。
2. 点击 同构遥操 。
3. 选择需要操作的配对。
4. 点击 启动同构遥操 开启功能。

5. 命令行启动与检查

5.1 环境准备

每个终端都先执行：

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash
```

说明：

- raybot_models 必须先加载，否则可能找不到 rb_a630_2_four_arms 模型资源。
- 启动硬件 bringup 的终端不要关闭。
- 零力拖动和同构遥操启动后，对应终端也不要关闭。

5.2 启动一对机械臂

本节以配对 1 为例： robot_1 主臂拖动， robot_3 从臂跟随。

终端 1：启动单对硬件 BRINGUP

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_homogeneous_teleop single_pair_homogeneous_teleop.launch.py \
  pair:=1 \
  start_bringup:=true \
  start_teleop:=false
```

等待出现：

```
Hardware ready state changed: READY
```

不要关闭终端 1。

终端 2：确认硬件和控制器

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 topic echo --once /raybot_robot/hardware_ready
ros2 control list_controllers
```

应看到：

```
data: true
```

控制器应包含：

```
joint_state_broadcaster        active
robot_1_arm_pos_controller     active
robot_1_arm_eff_controller     inactive
robot_3_arm_pos_controller     active
robot_3_arm_eff_controller     inactive
```

终端 3：启动 ROBOT_1 零力拖动

启动前托住主臂，并确保急停可用。

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_dynamics_control zero_force_drag_rb_a630_2.launch.py \
  robot_id:=robot_1
```

不要关闭终端 3。

终端 4：启动 ROBOT_1 到 ROBOT_3 遥操

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_homogeneous_teleop single_pair_homogeneous_teleop.launch.py \
  pair:=1 \
  start_bringup:=false \
  start_teleop:=true \
  master_effort_source:=external \
  follow_mode:=absolute
```

配对 2 的启动方式相同，将 `pair:=1` 改为 `pair:=2`，并将零力拖动改为：

```
ros2 launch raybot_dynamics_control zero_force_drag_rb_a630_2.launch.py \
  robot_id:=robot_2
```

配对 2 对应关系为： `robot_2` 主臂拖动， `robot_4` 从臂跟随。

5.3 启动四个机械臂

四臂模式会同时运行两路同构遥操：

- `robot_1` -> `robot_3`
- `robot_2` -> `robot_4`

终端 1：启动四臂硬件

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_homogeneous_teleop four_arms_homogeneous_teleop.launch.py \
```

```
  start_bringup:=true \
  start_teleop:=false
```

等待终端 1 启动完成，不要关闭该终端。

终端 2：检查硬件和控制器

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 topic echo --once /raybot_robot/hardware_ready
ros2 control list_controllers
```

必须看到：

```
data: true
```

终端 3：启动 ROBOT_1 零力拖动

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_dynamics_control zero_force_drag_rb_a630_2.launch.py \
  robot_id:=robot_1
```

终端 4：启动 ROBOT_2 零力拖动

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_dynamics_control zero_force_drag_rb_a630_2.launch.py \
  robot_id:=robot_2
```

终端 2：将两个主臂切换到 EFFORT

确保终端 3 和终端 4 已经正常运行，再执行：

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash
```

```
ros2 service call /raybot_robot/switch_robot_1_arm_to_effort \
  std_srvs/srv/Trigger "{}"

ros2 service call /raybot_robot/switch_robot_2_arm_to_effort \
  std_srvs/srv/Trigger "{}"
```

检查控制器：

```
ros2 control list_controllers
```

应看到以下控制器为 active ：

```
joint_state_broadcaster
robot_1_arm_eff_controller
robot_2_arm_eff_controller
robot_3_arm_pos_controller
robot_4_arm_pos_controller
```

此时先分别测试 robot_1 、 robot_2 零力拖动是否正常。确认主臂拖动手感和方向没有异常后，再启动同构遥操。

终端 5：启动两路同构遥操

```
cd /home/raybot/raybot_core_ws
source /opt/ros/humble/setup.bash
source /home/raybot/raybot_models/install/setup.bash
source install/setup.bash

ros2 launch raybot_homogeneous_teleop four_arms_homogeneous_teleop.launch.py \
  start_bringup:=false \
  start_teleop:=true \
  master_effort_source:=external \
  follow_mode:=absolute
```

5.4 停止流程

推荐按以下顺序停机：

1. 停止人工拖动主臂，确认四个机械臂均处于稳定状态。
2. 关闭同构遥操终端。
3. 关闭 robot_1 、 robot_2 零力拖动终端。
4. 关闭硬件 bringup 终端。
5. 再次确认机器人不再接收遥操或零力拖动力矩命令。

如需通过服务关闭零力拖动，可执行：

```
ros2 service call /zero_force_drag_robot_1/shutdown std_srvs/srv/Trigger "{}"
ros2 service call /zero_force_drag_robot_2/shutdown std_srvs/srv/Trigger "{}"
```

单对运行时，只关闭实际启动的主臂零力拖动节点即可。

6. 运行确认与成功判据

以下条件同时满足时，可认为同构遥操运行正常：

- /raybot_robot/hardware_ready 返回 data: true 。
- /joint_states 持续刷新。
- joint_state_broadcaster 为 active 。
- 主臂对应 eff_controller 为 active 。
- 从臂对应 pos_controller 为 active 。
- 主臂可被平稳小幅拖动，没有明显反向顶手、快速下落或抖动。
- 从臂跟随方向与主臂一致，且初始小幅动作验证通过。
- 启动遥操后没有持续出现状态超时、控制器切换失败或初始姿态误差过大的报错。

常用检查命令：

```
ros2 topic echo --once /raybot_robot/hardware_ready
ros2 control list_controllers
ros2 topic echo --once /joint_states
ros2 node list | grep homogeneous_teleop
ros2 node list | grep zero_force_drag
```

7. 常见问题与异常处理

问题现象	可能原因	处理方法	是否需立即停机
启动 bringup 时报已有 /controller_manager	Ray Studio 或其他终端已经启动机器人控制	先停止旧控制链路，再重新启动；不要同时运行两个 /controller_manager 。	否
/raybot_robot/hardware_ready 不是 true	硬件未就绪、电机未连接、控制组不匹配或 bringup 未完成	查看 bringup 终端日志，确认所需电机和控制组均已就绪。	否
零力拖动启动后主臂下沉或反向顶手	零力参数、重力补偿、关节状态或 effort 控制器状态异常	托住主臂并立即停止零力拖动，检查控制器状态和 /joint_states 。	是
同构遥操启动失败，提示初始位置误差过大	follow_mode:=absolute 下主从初始关节角差异超过安全阈值	先调整主从姿态到接近位置，再重新启动遥操。	否
从臂不跟随	遥操节点未启动、从臂 position 控制器未 active、 /joint_states 不完整或主臂状态超时	检查 ros2 node list 、 ros2 control list_controllers 和 /joint_states 。	否
从臂跟随方向或配对不符合预期	启动 pair 选择错误，或误解固定拓扑	确认 pair:=1 为 robot_1 -> robot_3 ， pair:=2 为 robot_2 -> robot_4 。	是
运行中从臂突然停止跟随并保持	主臂 /joint_states 超时或不完整	检查状态反馈链路，确认 /joint_states 持续刷新后再恢复操作。	是
停止后仍有节点存在	某个 launch 终端未关闭，或零力拖动节点仍在运行	使用 ros2 node list | grep -E 'homogeneous_teleop|zero_force_drag' 确认并关闭对应终端或服务。	否

8. 相关路径

项目	路径
同构遥操包	/home/raybot/raybot_core_ws/src/raybot_homogeneous_teleop
启动指令来源	/home/raybot/raybot_core_ws/src/raybot_homogeneous_teleop/doc/启动指令.md
同构遥操配置	/home/raybot/raybot_core_ws/src/raybot_homogeneous_teleop/config/homogeneous_teleop.yaml
同构遥操控制器配置	/home/raybot/raybot_core_ws/src/raybot_homogeneous_teleop/config/ros2_controllers.yaml
RB-A630-2 零力拖动 launch	/home/raybot/raybot_core_ws/src/raybot_dynamics_control/launch/zero_force_drag_rb_a630_2.launch.py
RB-A630-2 零力拖动配置	/home/raybot/raybot_core_ws/src/raybot_dynamics_control/config/zero_force_drag_rb_a630_2.yaml