

ARX采集-训练-部署手册

环境配置与代码编译

全套链路目前只适配ROS2系统

适配设备列表

产品型号	说明
AC one	\
X7s	\
LIFT2s	X5设备的上代LIFT同样兼容，R5版本的旧款LIFT不适配

模仿学习部分不负责技术售后，但全部迁移ACT进行开源。

注：本开源只让开发者能跑通整个流程。官方宣传中模型并非原版ACT，开源的代码达不到宣传视频中的泛化性以及流畅度，需开发者自行编写～



SDK下载

<https://github.com/ARXroboticsX>

X7s控制SDK	https://github.com/ARXroboticsX/X7s
X7s-模仿学习SDK	https://github.com/ARXroboticsX/ROS2_X7s_Play
LIFT控制SDK	https://github.com/ARXroboticsX/LIFT
LIFT模仿学习SDK	https://github.com/ARXroboticsX/ROS2_LIFT_Play
AC one控制SDK	https://github.com/ARXroboticsX/ARX_X5
AC one模仿学习SDK	https://github.com/ARXroboticsX/ROS2_AC-one_Play



代码结构


控制SDK与模仿学习仓库在同级目录下

环境配置

- 控制SDK

进入tools文件夹

依次执行如下文件


01_global_
nopasswd_
sudo.sh.x


02_apl_
update_
upgrade...


03_install_
common_
packages...

ROS2系统22.04建议选择humble，桌面版切记不要选择最小版本安装

ROS2系统24.04建议选择jazzy，桌面版切记不要选择最小版本安装


- IL环境配置


依次执行如下文件

每次执行前，新开启终端执行命令


01_install_
miniconda.
sh.x


02_config_
conda_
mirror.sh.x

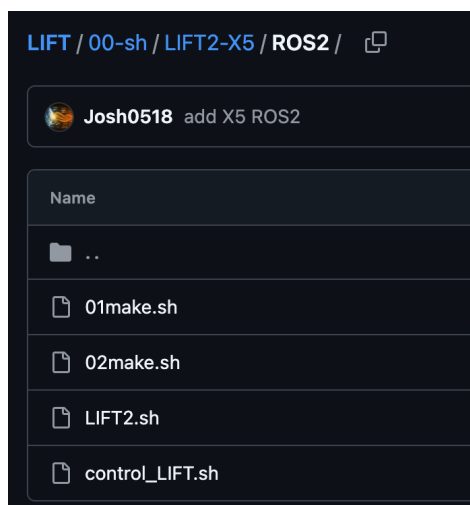

03_install_
act_env.sh.
x


04_install_
act_pip.sh.
x

注意：Ubuntu22.04需要删除~/miniconda/env/act/lib/中的libstdc++.6.0.29

SDK编译

进入对应控制部分SDK的00-sh文件夹,注意进入ROS2文件夹



执行01make.sh 等全部子窗口编译成功后，执行02make.sh

编译相机驱动

- realsense库安装

在主目录下

代码块

```
1  git clone https://github.com/IntelRealSense/librealsense.git
2
3  cd librealsense
4  mkdir build
5  cd build
6  cmake ../ -DBUILD_EXAMPLES=true
7  make -j$(nproc)
8  sudo make install
```

- 添加udev规则

在/home/arx/librealsense/ 目录下

代码块

```
1  sudo cp config/99-realsense-libusb.rules /etc/udev/rules.d/
2  sudo udevadm control --reload-rules && sudo udevadm trigger
```

- 依赖项安装

代码块

```
1  sudo apt install -y ros-$ROS_DISTRO-diagnostic-updater
2  sudo apt install -y ros-$ROS_DISTRO-image-transport-plugins
```

硬件配置

硬件连接

 AC one

<https://youtu.be/hpzz6BWvTGI>

依次将相机插入USB3.0插口即可，若通过hub连接，一个hub最多只能接入两个摄像头

注尽量保证一个USB口只有一个设备，防止多个设备互相干扰

- 相机连接

依次将相机插入USB3.0插口即可，若通过hub连接，一个hub最多只能接入两个摄像头

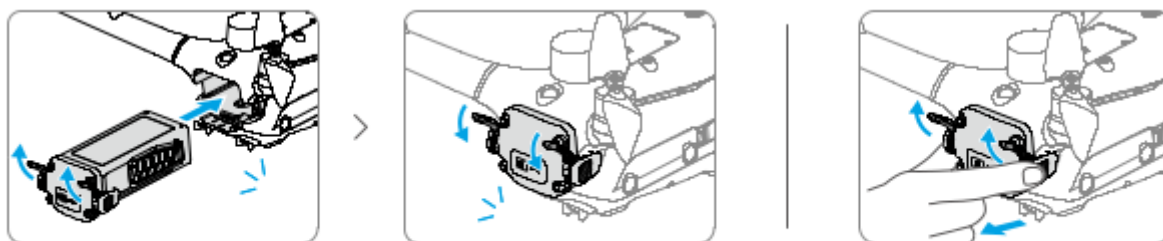
- VR连接



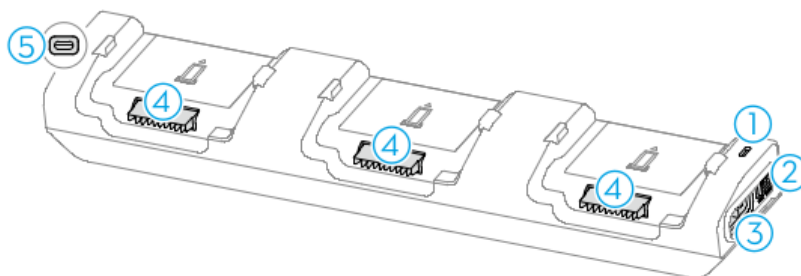
- CAN设备连接

将设备插入USB口即可

- 电池安装与充电

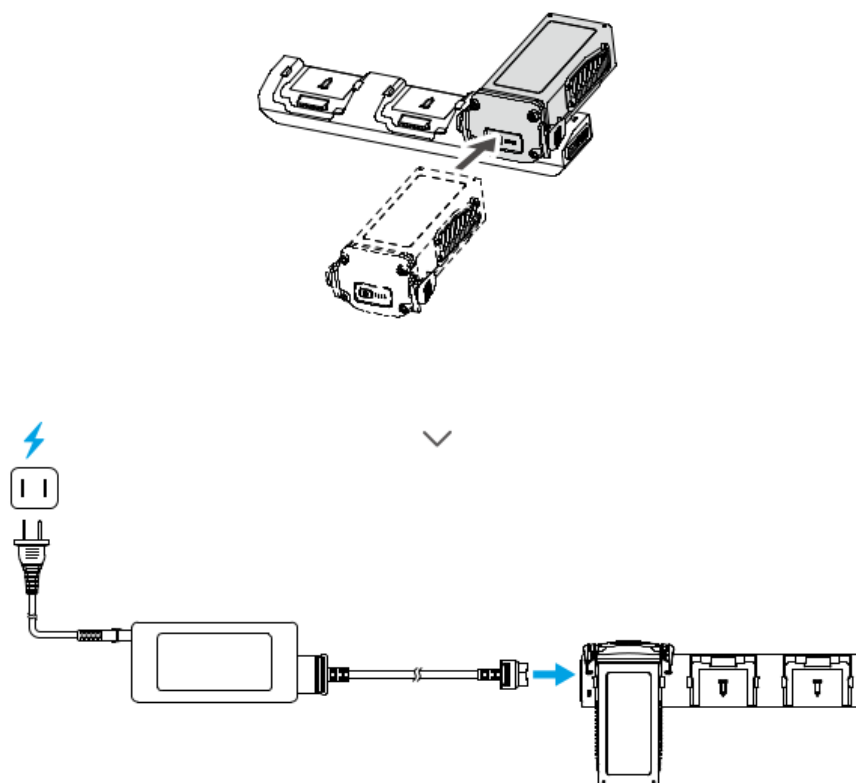


⚠ • 请勿在电源开启的情况下拆、装电池。




- | | |
|-----------|---------------|
| 1. 状态指示灯 | 4. 电池接口 |
| 2. 电源接口 | 5. USB-C 调参接口 |
| 3. 模式切换开关 | |

使用



拨动模式切换开关选择充电模式。

 **标准模式：**依次将每个电池充电至 100%。

 **待命模式：**依次将每个电池充电至 90%，便于快速使用电池。

充电管家将根据电池温度和电量依次为电池充电。充电耗时最短的电池将优先充电。

充电完成后，请取下电池并断开电源连接。

注意启动时将即停开关提前旋开

出现紧急情况用脚或工具将即停开关按下，注意断电时升降设备坠落

ARX信号设备配置

--	--	--	--	--

设备名称	左臂	右臂	机体	按键
X7s	can0	can1	can5	\
LIFT	can1	can3	can5	\
AC one	can1	can3	\	can6

四臂系统

左前	can1	右前	can3
左后	can0	右后	can2

LIFT 及 X7 出厂即配置无需更改

相机配置

在电脑上连接realsense d405的usb

ROS2_XX_Play/realsense目录下

代码块

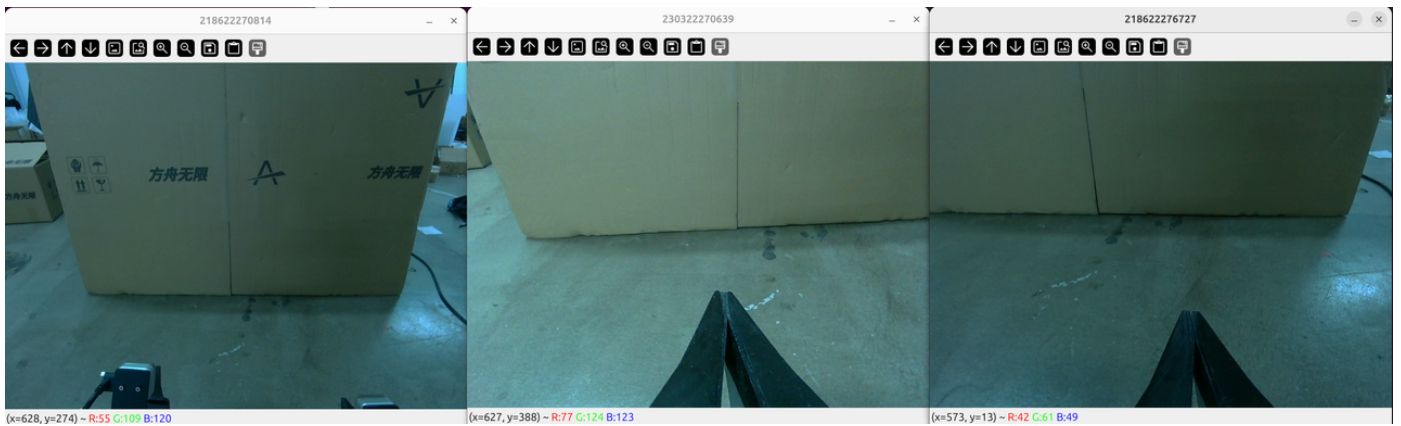
```
1 colcon build
```

代码块

```
1 ./serach.sh
```

若显示出的USB口不为3.2，请检查相机连接是否正常,确保都是3.2输出

```
$ ros2 run realsense2_camera list_camera_node
Found 3 device(s):
- Serial: 218622276727 , USB: 3.2
- Serial: 230322270639 , USB: 3.2
- Serial: 218622270814 , USB: 3.2
Press any key in the image window(s) to exit...
```



显示的图像对应的标题及为其序列号，根据图像判断左右和头部，将获取到的Serial number填入到 `realsense.sh` 中

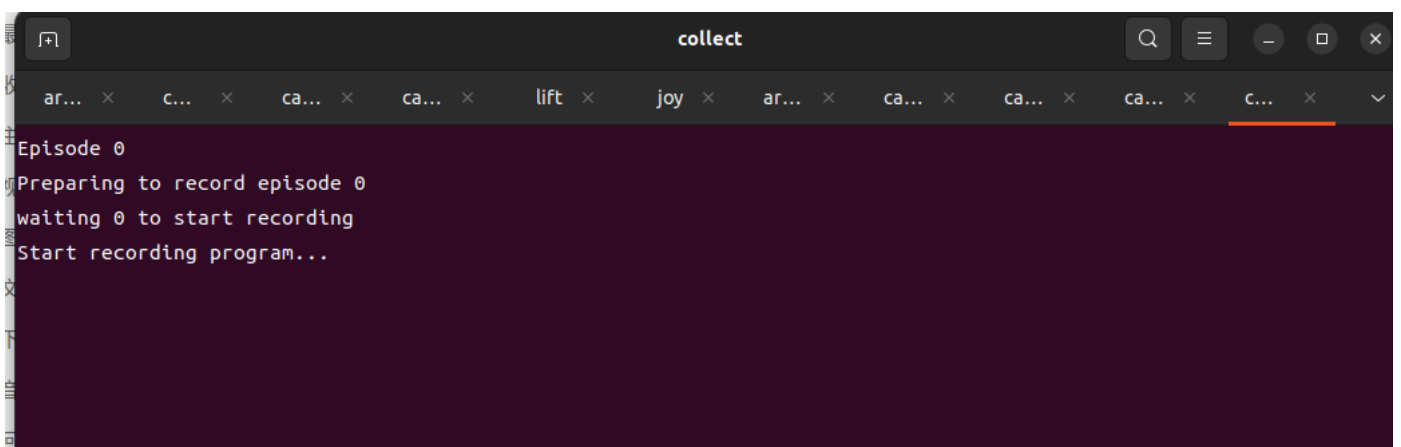
```
declare -A CAMS=(  
  [camera_h]="218622270814"  
  [camera_l]="218622276727"  
  [camera_r]="230322270639"  
)
```

确保相机序列号与安装的对应，从上至下依次为头部，左臂，右臂

采集 | 遥操作

执行XX_Play/tools/ 目录下的 01_collect.sh

出现如下代表采集正常



AC one

初次启动后按2 机械臂到达期望位置后；

按1 开始采集，电脑语音提示Go。

采集后按2进行自动复位并保存，电脑语音提示Safe。

LIFT2s&X7s

开始采集：长按左右臂复位，电脑语音提示Go。

保存数据：长按左右臂复位，复位后自动保存，同时电脑语音提示Safe。

成功采集后在如下目录下即可看到产生的数据。



VR遥操作介绍

<https://youtu.be/-9LXCBwEN0s>

训练

执行XX_Play/tools/ 目录下 02_train.sh

出现如下进度条即代表训练正常

```
ROBOMIMIC WARNING(
  No private macro file found!
  It is recommended to use a private macro file
  To setup, run: python /home/arx/AC_one/ac_one_play_ros2/act/robomimic/scripts/setup_macros.py
)
args.camera_names=['head', 'left_wrist', 'right_wrist']
Detect 6 episodes in dataset directory
states_dim=14 action_dim=28

Data from: /home/arx/AC_one/ac_one_play_ros2/act/datasets

number of parameters: 106.26M
0%|                                     | 0/3000 [00:00<?, ?it/s]
Saved plots to /home/arx/AC_one/ac_one_play_ros2/act/weights
2%|█                                   | 50/3000 [00:40<33:34, 1.46it/s]
```

训练结束后会出现类似如下文件



部署

执行XX_Play/tools/ 目录下 03_inference.sh

在inference目录下按下回车，即开始自动推理

```
ROBOMIMIC WARNING(  
  No private macro file found!  
  It is recommended to use a private macro file  
  To setup, run: python /home/arx/ros2-x7s/ROS2_X7s_Play/act/robomimic/scripts/setup_macros.py  
)  
action_dim=32 states_dim=16  
follow_arm_publish_continuous: 1  
follow_arm_publish_continuous: 2  
follow_arm_publish_continuous: 3  
follow_arm_publish_continuous: 4  
follow_arm_publish_continuous: 5  
follow_arm_publish_continuous: 6  
follow_arm_publish_continuous: 7  
follow_arm_publish_continuous: 8  
follow_arm_publish_continuous: 9  
follow_arm_publish_continuous: 10  
follow_arm_publish_continuous: 11  
follow_arm_publish_continuous: 12  
follow_arm_publish_continuous: 13  
follow_arm_publish_continuous: 14  
follow_arm_publish_continuous: 15  
follow_arm_publish_continuous: 16  
follow_arm_publish_continuous: 17
```

采集及部署技巧

注意以下操作更改后需重新编译代码，注意代码保存后编译

高度控制

针对LIFT2s 及 X7s

- 采集时固定高度

该变量对应范围为0-20即setHeight(0/41.54)到setHeight(20/41.54)

XX/body/ROS2/src/ARX_LIFT_ros2/arx_lift_controller/src/lift_controller.cpp

更改ARX_VR_L接收的话题，将height改为固定变量

```
45   ros::Subscriber sub = nh.subscribe<arm_control::PosCmd>(
46       "/ARX_VR_L", 1, [&](const arm_control::PosCmd::ConstPtr &msg) {
47           collect = true;
48           // control_loop.setHeight(msg->height / 41.54);
49           control_loop.setHeight(5 / 41.54);
50           control_loop.setWaistPos(msg->tempFloatData[0]);
51           double yaw, pitch;
```

- 推理时固定高度

XX/body/ROS2/src/ARX_LIFT_ros2/arl_lift_controller/src/lift_controller.cpp

body_control话题下注释设定高度值

```
ros::Subscriber body_control_sub = nh.subscribe<arm_control::PosCmd>(
    "/body_control", 1, [&](const arm_control::PosCmd::ConstPtr &msg) {
        collect = false;
        last_cb_time = ros::Time::now();
        // control_loop.setHeight(msg->height / 41.54);
        control_loop.setWaistPos(msg->tempFloatData[0]);
        control_loop.setHeadYaw(msg->head_yaw);
        control_loop.setHeadPitch(-msg->head_pitch);
        control_loop.setWheelVel(msg->tempFloatData[1], msg->tempFloatData[2],
                                msg->tempFloatData[3], msg->tempFloatData[4]);
        control_loop.setChassisCmd(0, 0, 0, msg->model);
    });
```

在主函数中设定目标高度

```
111   while (ros::ok()) {
112       control_loop.loop();
113
114       control_loop.setHeight(5 / 41.54);
115   }
```

头部控制

针对LIFT2s 及 X7s

XX/body/ROS2/src/ARX_LIFT_ros2/arl_lift_controller/src/lift_controller.cpp

在主函数中，调用此两个函数进行姿态设置，单位为rad

```
85   while (rclcpp::ok()) {
86       control_loop->loop();
87       control_loop->setHeadYaw(0);
88       control_loop->setHeadPitch(0.5);
89   }
```

底盘速度反馈

针对LIFT2s 及 X7s

反馈话题

/body_information

底盘速度控制

针对LIFT2s 及 X7s

控制话题

/body_control