**TRON1 RL训练部署快速上手（Isaac Gym）**

|  |
| --- |
| 本文主要介绍如何快速使用逐际动力的开源强化学习仓库实现TRON1的训练和部署 |

1. **安装与环境配置**

1.1 **前置条件**

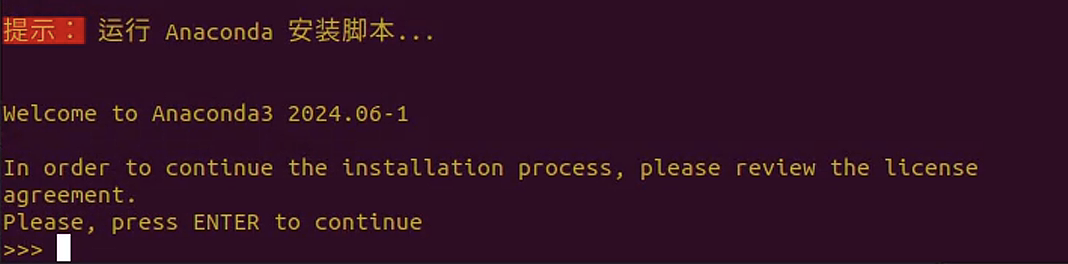
1. GPU：推荐 GeForce RTX 3080 (12 GB) 或更高
2. 操作系统：Ubuntu 20.04 LTS
3. 网络环境：能正常访问Github

1.2 **一键安装RL训练环境**

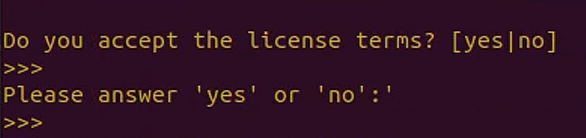
在终端中输入：

|  |
| --- |
| bash mkdir -p ~/limx\_rl && cd ~/limx\_rl && \ sudo apt update && sudo apt install -y git && \ if [ ! -d "pointfoot-legged-gym" ]; then \  git clone https://github.com/limxdynamics/pointfoot-legged-gym.git; \ fi && \ cd pointfoot-legged-gym && bash install.sh && source ~/.bashrc |

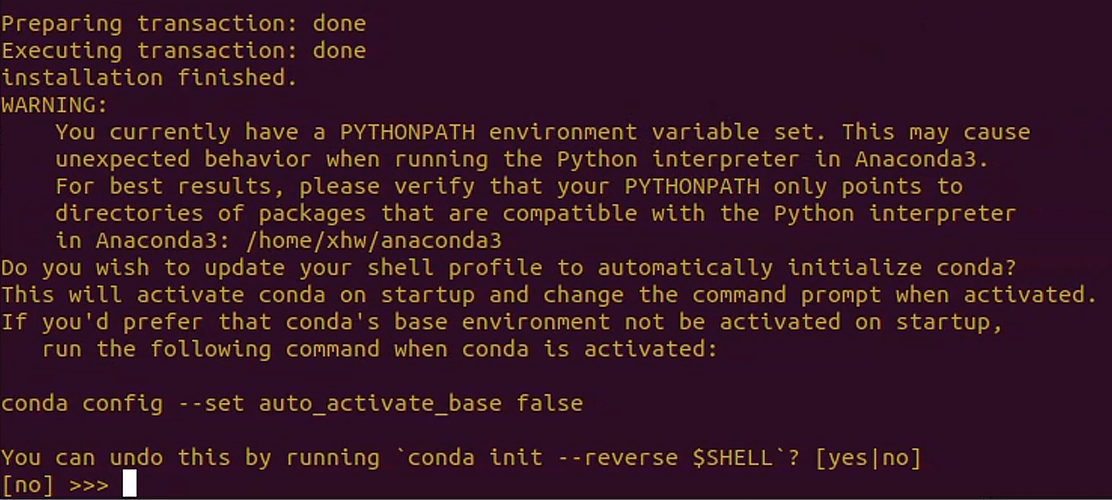
执行完上述命令后等待片刻，终端中会出现：



此时长按Enter阅读完Anaconda的用户协议后，会出现：



输入yes并回车后，可以选择Anaconda的路径（一般不用更改），直接按下Enter即可，会出现：



输入no按下回车即可完成Anaconda的安装。

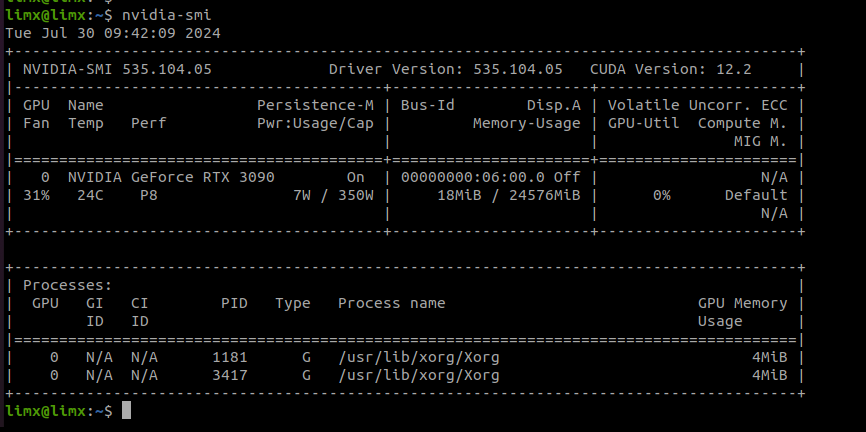
Anaconda安装完成后，一键安装脚本还会继续安装pointfoot\_legged\_gym和相关依赖，请耐心等待安装完成。

1.3 **验证训练环境**

1. 验证Nvidia驱动是否安装成功

|  |
| --- |
| Plain Text nvidia-smi |

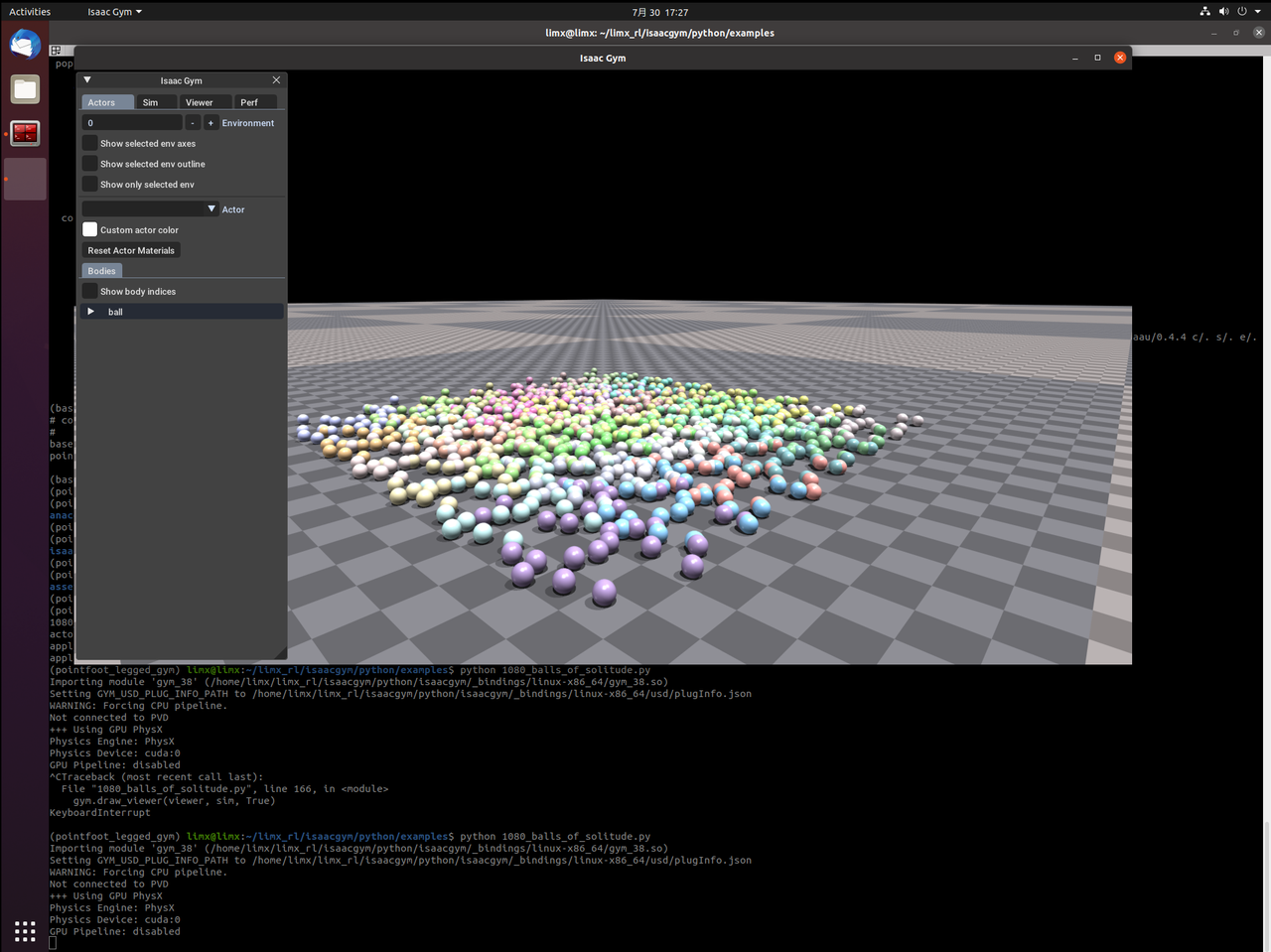
如果显示了 NVIDIA 驱动版本和 GPU 信息，说明驱动已成功安装。



1. 验证Isaac Gym是否安装成功

运行Isaac Gym 的示例，它可以帮助确认 Isaac Gym 是否正确安装并配置。

|  |
| --- |
| Plain Text conda activate pointfoot\_legged\_gym cd ~/limx\_rl/isaacgym/python/examples python 1080\_balls\_of\_solitude.py |



1. 验证pointfoot-legged-gym是否安装成功

当您安装完RL环境后，在 ~/limx\_rl目录下包含如下所示的内容：

|  |
| --- |
| Plain Text cd ~/limx\_rl . ├── isaacgym └── pointfoot-legged-gym |

1.4 **一键安装部署环境**

打开一个终端，在终端中输入（注意训练的conda环境和部署的conda环境不是同一个）：

|  |
| --- |
| bash source ~/anaconda3/etc/profile.d/conda.sh && \ conda create -n pointfoot\_deploy python=3.8 -y && \ conda activate pointfoot\_deploy && \ mkdir -p ~/limx\_ws && cd ~/limx\_ws && \ git clone --recurse-submodules https://github.com/limxdynamics/pointfoot-mujoco-sim.git && \ git clone --recurse-submodules https://github.com/limxdynamics/rl-deploy-with-python.git && \ ARCH=$(uname -m) && \ if [ "$ARCH" = "x86\_64" ]; then \  pip install rl-deploy-with-python/limxsdk-lowlevel/python3/amd64/limxsdk-\*-py3-none-any.whl; \ elif [ "$ARCH" = "aarch64" ]; then \  pip install rl-deploy-with-python/limxsdk-lowlevel/python3/aarch64/limxsdk-\*-py3-none-any.whl; \ else \  echo "不支持的架构：$ARCH"; \ fi && \ pip install onnx && \ echo 'export ROBOT\_TYPE=PF\_TRON1A'>>~/.bashrc && \ echo'export RL\_TYPE=isaacgym'>> ~/.bashrc && \ source ~/.bashrc |

1.5 **验证部署环境**

打开一个Bash终端来运行控制算法。

|  |
| --- |
| Bash # 激活pointfoot\_deploy的conda环境 conda activate pointfoot\_deploy  # 运行控制算法 cd ~/limx\_ws && python rl-deploy-with-python/main.py |

打开一个Bash终端来运行MuJoCo仿真器。

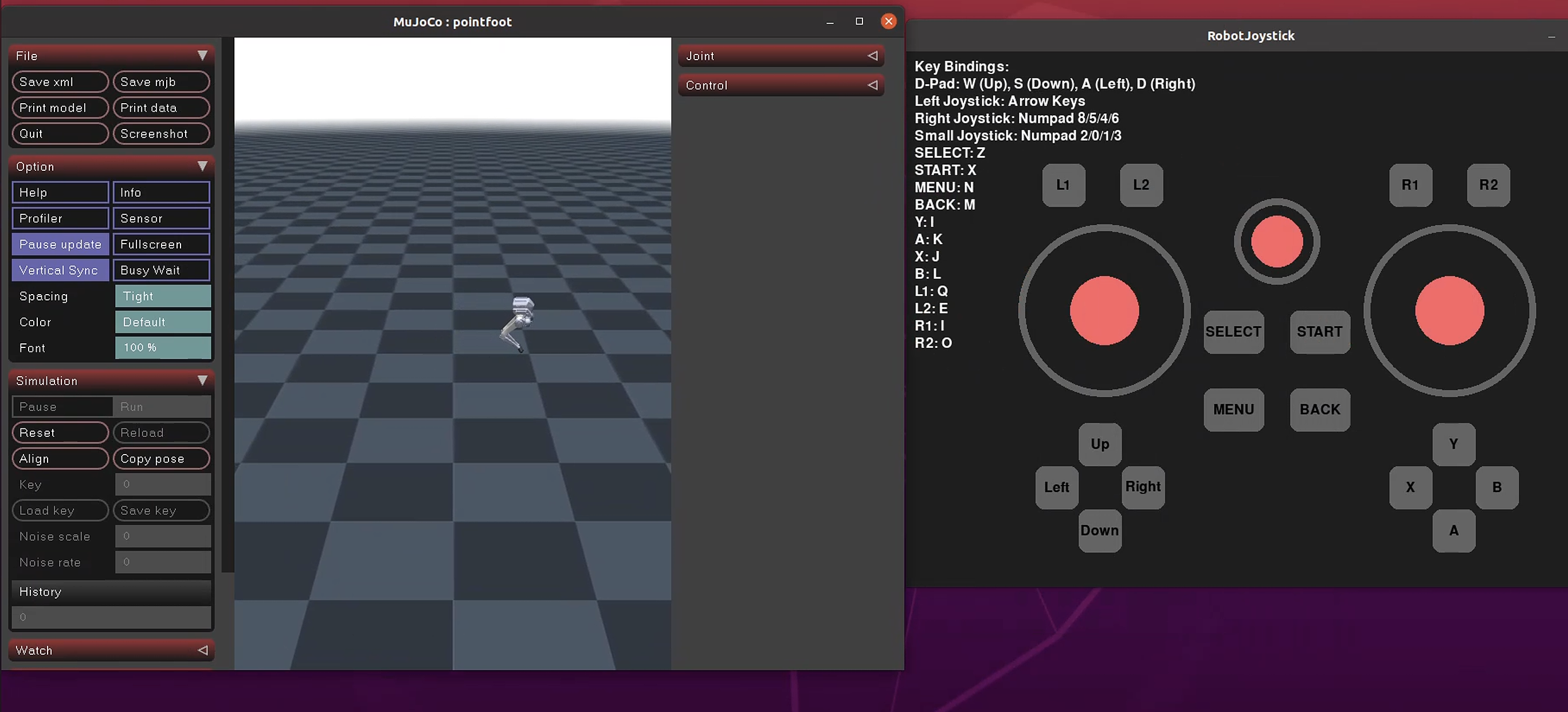
|  |
| --- |
| Bash # 激活pointfoot\_deploy的conda环境 conda activate pointfoot\_deploy  # 运行仿真器 cd ~/limx\_ws && python pointfoot-mujoco-sim/simulator.py |

打开一个Bash终端来运行虚拟遥控器，仿真的过程中，您可以使用虚拟遥控器来控制机器人运动。

* 左摇杆：控制前进/后退/左转/右转运动；
* 右遥控：可控制机器人左右横向运动。

|  |
| --- |
| Plain Text cd ~/limx\_ws && ./pointfoot-mujoco-sim/robot-joystick/robot-joystick |

若能看到双点足机器人在MuJoCo中正常运动说明配置成功。



如果您的机器人处于摔倒状态，请在启动运动控制后，单击 MuJoCo 仿真器左侧菜单栏的"Reset"按钮以重置机器人。此时，您将看到机器人恢复并开始行走。

2. **TRON1点足训练**

2.1 **启动训练**

1. 训练双点足机器人需切换分支，打开终端输入：

|  |
| --- |
| Plain Text cd ~/limx\_rl/pointfoot-legged-gym |

1. 常用训练相关参数指令说明：

* --task=pointfoot\_flat：指定任务或环境类型为 pointfoot\_flat。
* --num\_envs 1024：指定要创建的环境数量为 1024。
* --max\_iteration 10000: 指定要训练的轮数为10000轮。
* --headless：以无头模式运行，即不显示任何图形界面。通常用于在没有显示器的服务器上运行，或者在需要高效计算而不需要图形渲染的情况下使用。

1. 开始训练：
2. 命令1：无头模式运行（推荐使用）

|  |
| --- |
| Bash conda activate pointfoot\_legged\_gym cd ~/limx\_rl/pointfoot-legged-gym python legged\_gym/scripts/train.py --task=pointfoot\_flat --headless |

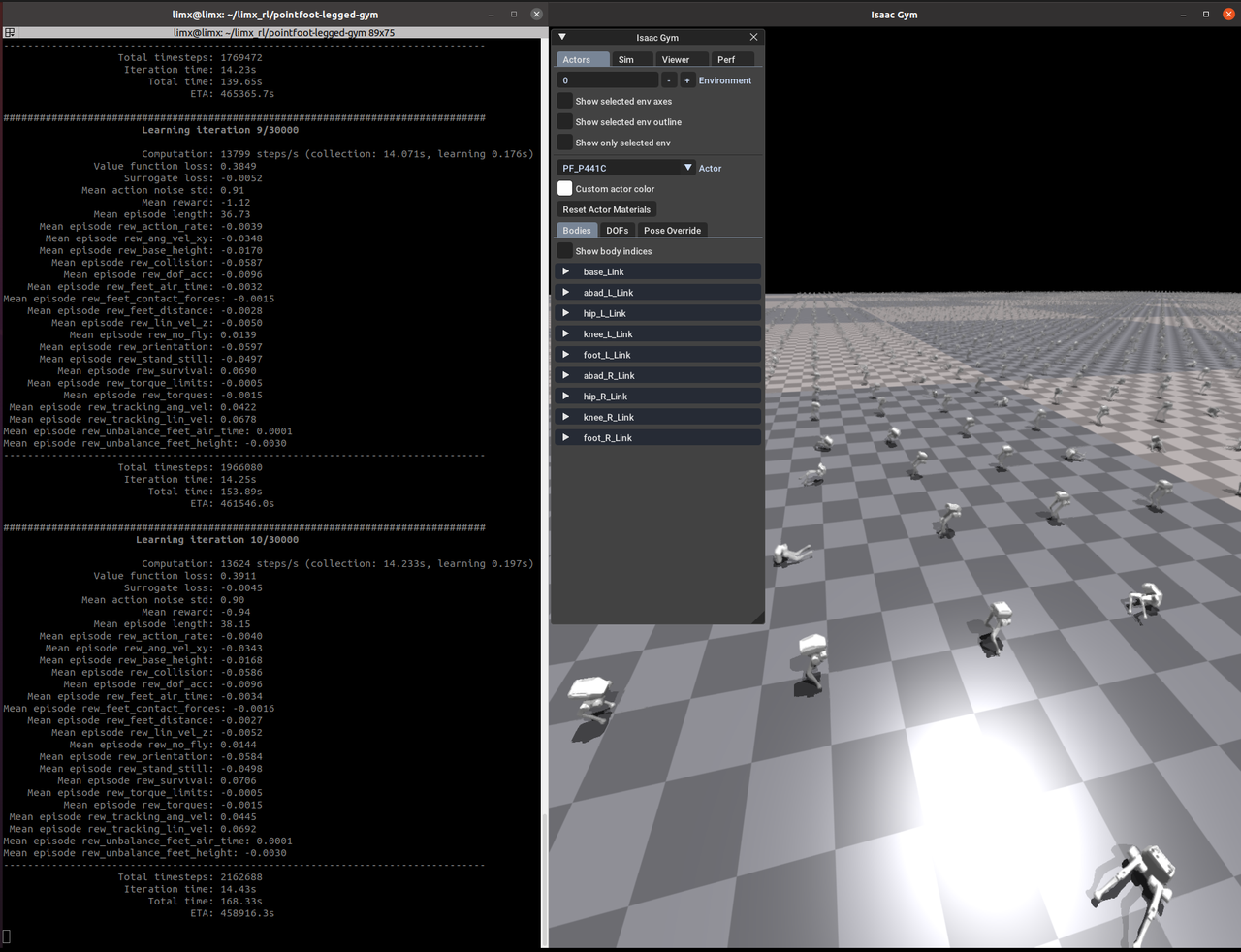
此时训练以无图形化运行，由于不需要图形渲染，训练速度较快。

1. 命令2：图形模式运行

|  |
| --- |
| Bash conda activate pointfoot\_legged\_gym cd ~/limx\_rl/pointfoot-legged-gym python legged\_gym/scripts/train.py --task=pointfoot\_flat |

可以看到Isaac Gym以图形模式运行：

（可以在键盘按下V键暂停图形渲染来加快训练速度）



1. 继续训练：

如果您的训练终止，可以指定一个检查点的文件接着继续训练。

请注意将--load\_run和--checkpoint的参数替换为实际中您的参数。

|  |
| --- |
| Bash cd ~/limx\_rl/pointfoot-legged-gym python legged\_gym/scripts/train.py --task=pointfoot\_flat --resume --headless --load\_run Dec23\_17-38-22\_ --checkpoint 200 |

可以看到Loading model from: 的输出，说明已经读取该路径下的checkpoint继续训练。



2.2 **查看训练情况**

1. 激活conda环境

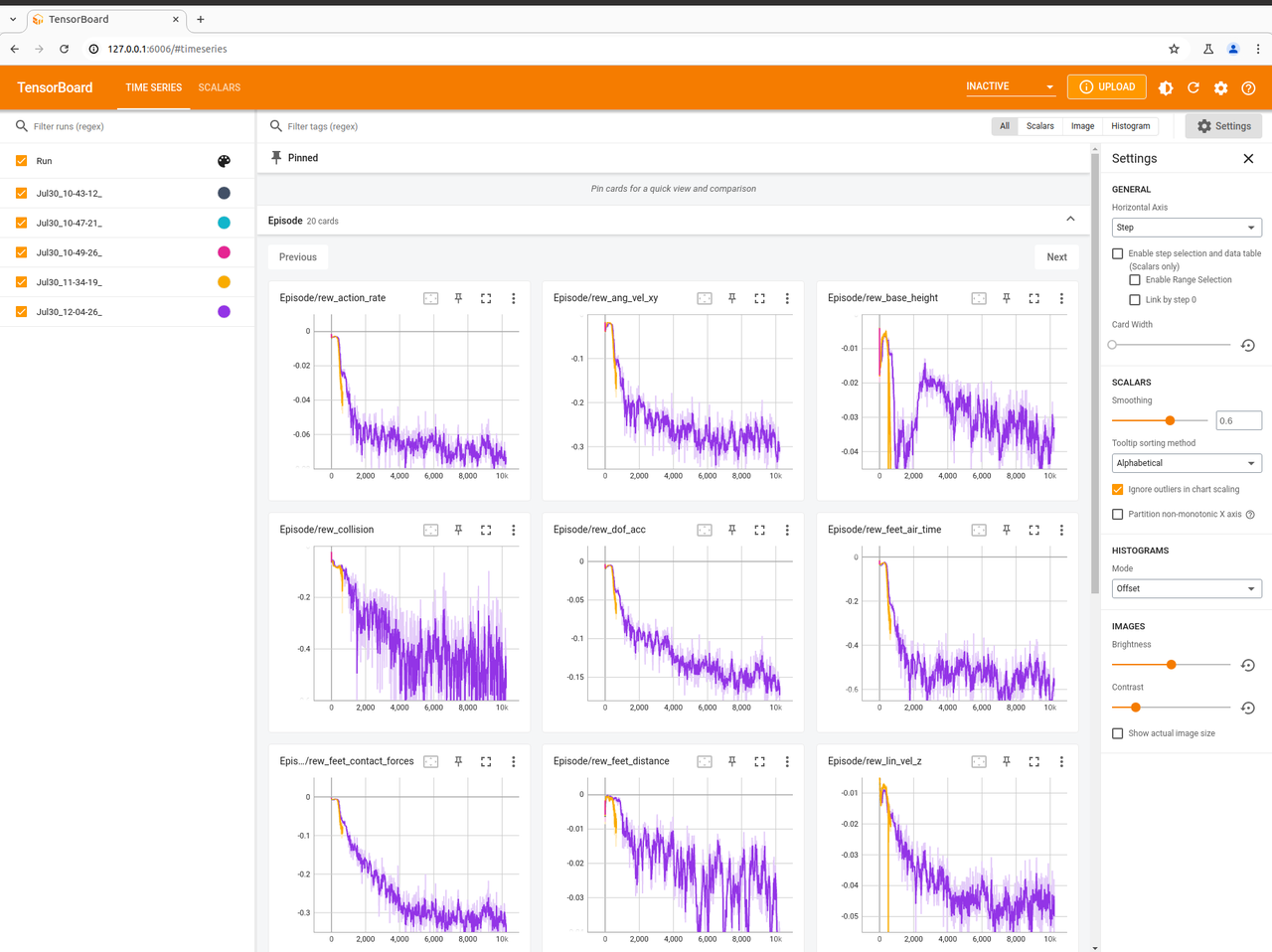
|  |
| --- |
| Plain Text conda activate pointfoot\_legged\_gym |

1. 启动 tensorboard

|  |
| --- |
| Plain Text cd ~/limx\_rl/pointfoot-legged-gym tensorboard --logdir=logs/pointfoot\_flat |

1. 查看训练情况

在浏览器地址栏输入 http://127.0.0.1:6006，查看训练情况。



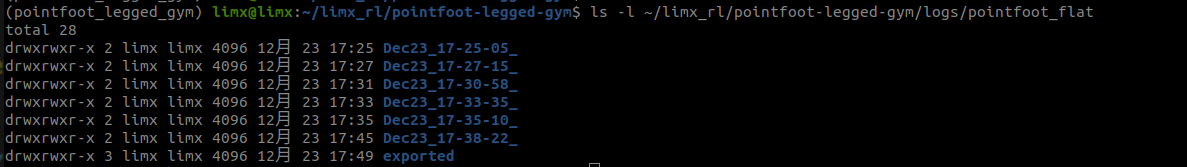
2.3 **导出训练结果**

1. 完成训练后查看训练结果

* 默认会读取最新的run和checkpoint，如需选择特定的run和checkpoint，请输入--load\_run和--checkpoint参数。
* --load\_run:：指定要加载的训练运行的标识符（例如，训练运行的名字或ID）。这个标识符通常与训练过程相关联，用于从 logs 目录中找到相应的运行记录或配置。
* 获取方式：查看 logs 目录：进入 logs 目录查看子目录或文件，这些子目录或文件通常会以训练运行的标识符命名。
* 示例路径：

|  |
| --- |
| Plain Text ls -l ~/limx\_rl/pointfoot-legged-gym/logs/pointfoot\_flat |

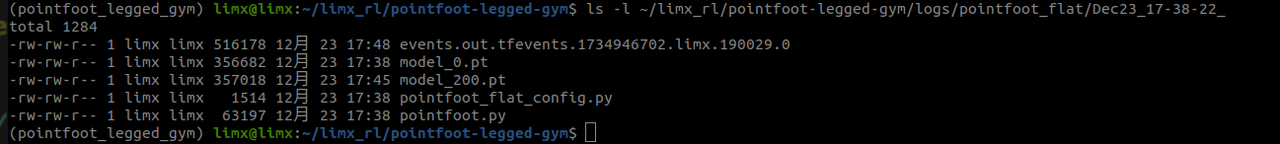
您会看到类似于 Dec23\_17-38-22\_ 这样的目录，Dec23\_17-38-22\_ 就是 --load\_run 参数的值。



* --checkpoint：指定要加载的检查点文件。检查点文件保存了模型的中间状态，可以用于恢复训练或进行推断。
* 获取方式：查看 logs 目录：在 logs 目录下的相应 --load\_run 目录中，通常会有保存检查点的文件。这些文件通常以 .pt 或类似扩展名存在，文件名可能包含训练的轮次或时间戳。
* 示例路径：

|  |
| --- |
| Plain Text ls -l ~/limx\_rl/pointfoot-legged-gym/logs/pointfoot\_flat/Dec23\_17-38-22\_ |

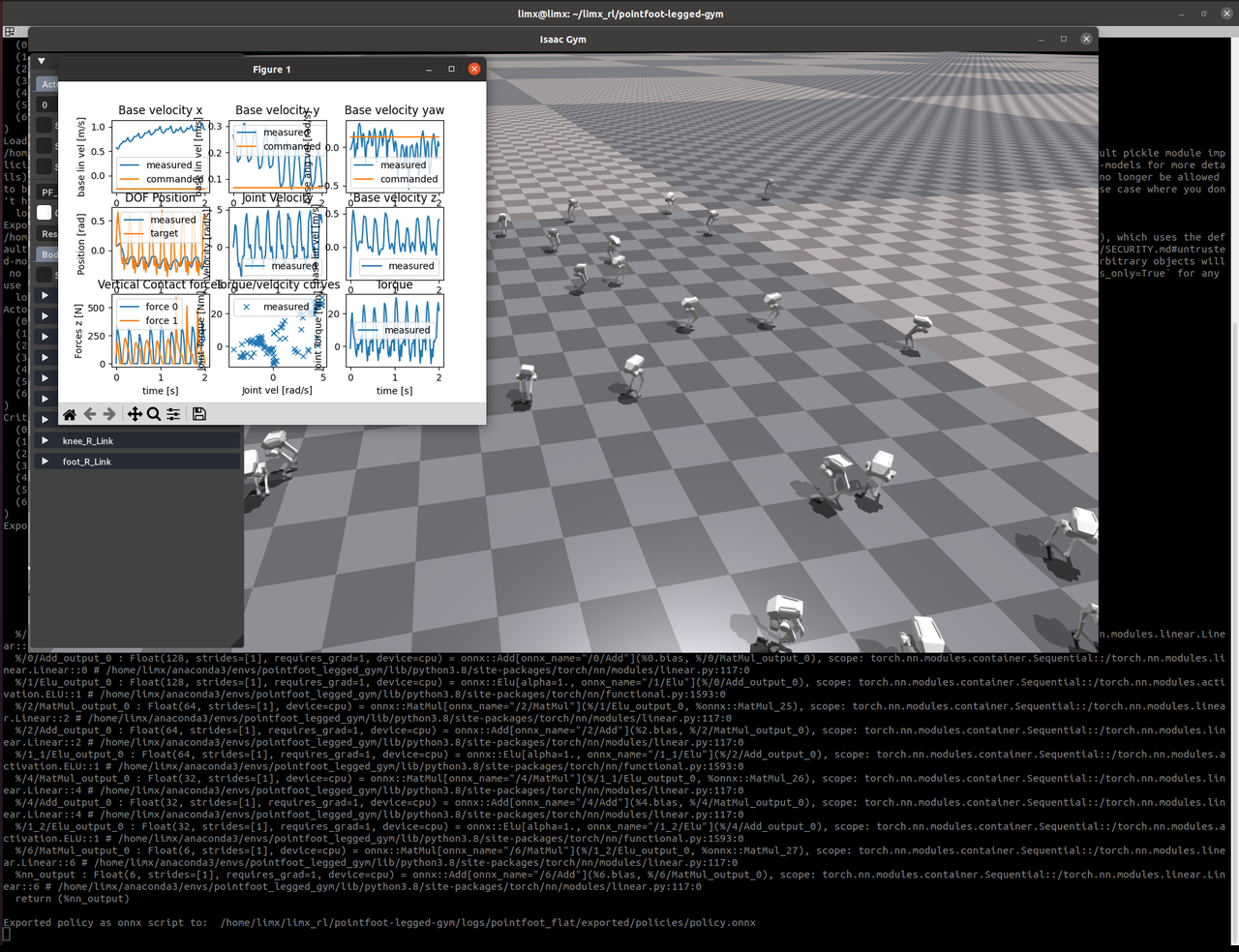
您会看到类似于model\_200.pt 的文件，则200就是 --checkpoint 参数的值。



* 使用示例

请注意将--load\_run和--checkpoint后的参数替换为自己训练logs 目录中的。

|  |
| --- |
| Plain Text conda activate pointfoot\_legged\_gym cd ~/limx\_rl/pointfoot-legged-gym python legged\_gym/scripts/play.py --task=pointfoot\_flat --load\_run Dec23\_17-38-22\_ --checkpoint 200 |

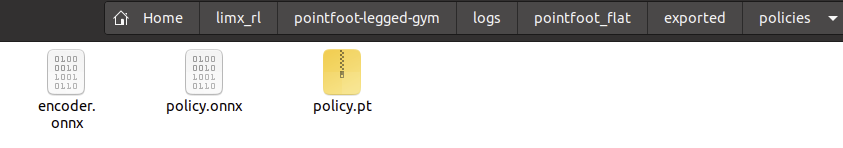


1. 导出模型的ONNX格式文件

* 在运行完上一步的脚本后，可以到目录：

~/limx\_rl/pointfoot-legged-gym/logs/pointfoot\_flat/exported/policies 中查看导出的文件。

|  |
| --- |
| Plain Text ls -l ~/limx\_rl/pointfoot-legged-gym/logs/pointfoot\_flat/exported/policies |



2.4 **基于Python仿真部署（sim2sim）**

1. 在您的工作空间中，以 PF\_TRON1A机器人类型为例，RL模型和配置文件所在路径为：

~/limx\_ws/rl-deploy-with-python/controllers/model/PF\_TRON1A，如下所示。

|  |
| --- |
| Plain Text tree ~/limx\_ws/rl-deploy-with-python/controllers/model/PF\_TRON1A . ├── params.yaml └── policy  └── policy.onnx  └── encoder.onnx |

1. 将导出的训练结果即ONNX文件（包括policy.onnx和encoder.onnx）移动到对应的policy文件夹内即可。
2. 启动仿真

打开一个Bash终端来运行控制算法。

|  |
| --- |
| Plain Text conda activate pointfoot\_deploy cd ~/limx\_ws && python rl-deploy-with-python/main.py |

打开一个Bash终端来运行MuJoCo仿真器。

|  |
| --- |
| Plain Text conda activate pointfoot\_deploy cd ~/limx\_ws && python pointfoot-mujoco-sim/simulator.py |

打开一个Bash终端来运行虚拟遥控器。

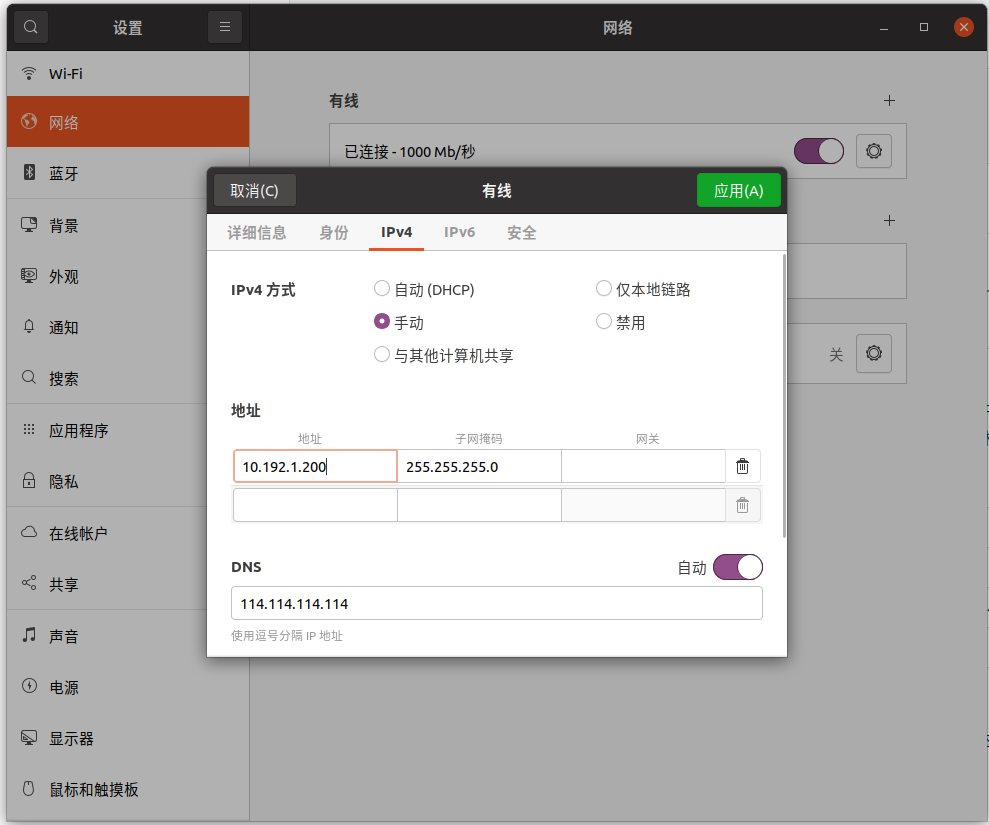
|  |
| --- |
| Plain Text cd ~/limx\_ws && ./pointfoot-mujoco-sim/robot-joystick/robot-joystick |

即可看到训练结果部署在MuJoCo平台中的表现。

3. **TRON1实机部署**

3.1 **真机调试**

1. 修改开发者电脑IP：确保您的开发电脑与机器人本体通过外置网口连接。设置您的电脑IP地址为：10.192.1.200，并通过Shell命令ping 10.192.1.2 能够正常ping通。如下图所示对您的开发电脑进行IP设置：



1. 打开开发者模式：机器人开机后，同时按下遥控器按键 R1 + Left，这时本体主机将会自动重启并切换机器人到开发者模式。在此模式下，用户可以开发自己的运动控制算法。模式设置掉电后不会失效，重新开机后仍然是开发者模式。
2. 进行校零动作：机器人开机启动后，执行运控程序之前，请进行校零，使机器人各个关节回到初始位置。校零对应的遥控器按键为L1+R1。
3. 实机部署运行。在Bash终端只需下面Shell命令启动控制算法（**在进行实机部署运行时，确保机器人吊装非常重要**）：

|  |
| --- |
| Plain Text conda activate pointfoot\_deploy cd ~/limx\_ws python rl-deploy-with-python/main.py 10.192.1.2 |

* 这时您可以通过遥控器按键L1 + △开启机器人行走功能。左摇杆：控制前进/后退/左转/右转运动；右遥控：可控制机器人左右横向运动。
* 遥控器按L1 + □关闭机器人行走功能。

3.2 **真机部署**

当您完成仿真和真机调试后，可以将您的算法程序部署到机器人上**（在部署测试完成之前，确保机器人始终保持吊装状态非常重要，以保障安全）**。详细步骤如下：

1. 准备工作

* 保持机器人继续处于开发者模式：确保机器人仍处于开发者模式，方便您进行程序部署和调试。
* 网络连接：确保开发电脑与机器人通过外置网口（Ethernet）连接，网络稳定且通信正常。部署完成后，网络连接将不再需要。

1. 拷贝算法程序到机器人

* 在开发电脑上打开终端，进入到存放算法程序的工作目录，例如 ~/limx\_ws。
* 使用 scp 命令将包含算法的目录拷贝到机器人中，默认机器人用户为 guest，密码为 123456。

|  |
| --- |
| Plain Text cd ~/limx\_ws scp -r rl-deploy-with-python guest@10.192.1.2:/home/guest |

* 安装运动控制开发库（如果尚未安装）：

|  |
| --- |
| Plain Text ssh guest@10.192.1.2 "pip install /home/guest/rl-deploy-with-python/limxsdk-lowlevel/python3/amd64/limxsdk-\*-py3-none-any.whl" |

* 配置算法自动启动：

SSH 登录机器人：使用 ssh 命令远程登录到机器人系统，密码为 123456。

|  |
| --- |
| Plain Text ssh guest@10.192.1.2 |

打开自启动脚本 /home/guest/autolaunch/autolaunch.sh 进行编辑：

|  |
| --- |
| Plain Text busybox vi /home/guest/autolaunch/autolaunch.sh |

在脚本中找到启动 main.py 的命令：确保 python3 /home/guest/rl-deploy-with-python/main.py 10.192.1.2 这一行未被注释（没有 # 注释符号）。编辑完成后，保存并退出编辑器。

|  |
| --- |
| bash #!/bin/bash  while true; do  # 启动用于控制 Pointfoot 机器人的 Python 控制器脚本  # 参数 10.192.1.2 表示机器人所在的 IP 地址  # 该行被注释掉，若要启用自动控制器启动，请取消注释  # 请根据您控制器脚本的实际路径修改此路径  python3 /home/guest/rl-deploy-with-python/main.py 10.192.1.2   # 使用 roslaunch 启动机器人控制算法  # 若要启动机器人控制算法，请取消下面注释  # 请根据您安装的实际路径修改路径  # source /home/guest/install/setup.bash  # roslaunch robot\_hw pointfoot\_hw.launch   # 等待 3 秒后重新启动  sleep 3 done |

* 重启机器人电脑。配置完成后，手动重启机器人。

1. 控制机器人运动

等待系统启动后，您可以使用遥控器控制机器人：

* L1 + △：启动机器人行走功能；
* L1 + □： 关闭机器人行走功能；
* 左摇杆：控制机器人前进、后退、左转、右转；
* 右摇杆：控制机器人左右横向运动。