最低显卡要求说明：建议至少 12–16 GB 显存的 NVIDIA GPU（3080Ti及以上）。原因：所用大模型权重 `pytorch_model.bin` 约 8.4 GB，加载到 GPU 后还需额外显存存放 KV cache、激活缓冲与系统预留空间； 显存不足会导致加载失败或运行时 OOM。

注意：本例程使用的是GNSS来进行绝对定位，未使用视觉SLAM之类定位数据。优势：算力要求相对较低，飞行仿真更稳定；缺点：和真机情况不是特别一致，真实情况下室内无GNSS卫星信号，通常需要使用视觉或激光SLAM，或直接使用VIO模块来进行相对定位。若要使用SLAM定位方式，请参考6.LLMUavCompSLAM实验。

1.环境准备

由于WinWSL开发环境为了节省体积，对部分环境进行了精简。本实验需要使用RflySim基于WSL2/Ubuntu 22.04开发的**外挂环境WinWSL2-GPU**来运行本实验。本外挂环境，附带了docker、cuda、pytorch、tensorflow等额外功能，能够开发更复杂的无人系统感知决策算法。

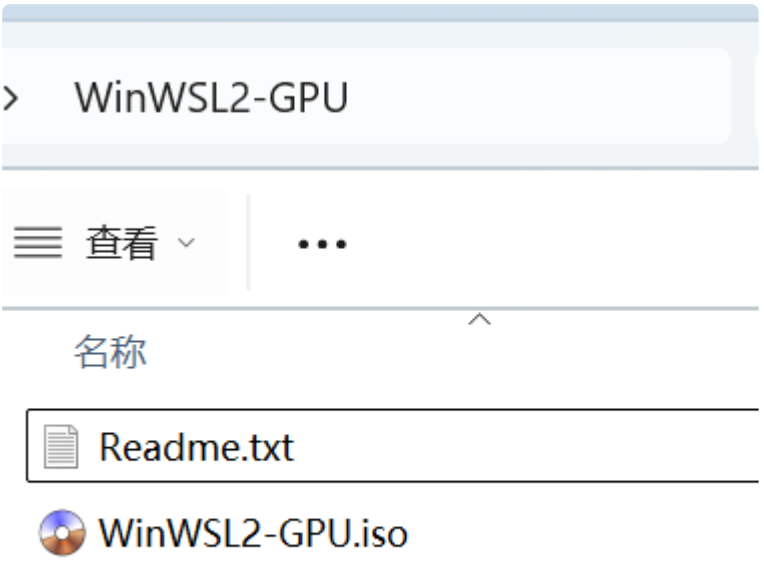
注意：本实验支持Windows10+Windows11 （22H2之下和之上版本都支持）。

注意：RflySim的WinWSL2-GPU编译器理论上需要Windows11 22H2及以上系统才可以直接使用。之前的系统理论上需要配置一些IP，才能实现WinWSL2-GPU和Windows与WSL上其他程序通信，不过本例程是用了自动请求IP的机制，因此可以直接运行。

1.1 WSL2-GPU增量环境包下载

可根据实验例程：`1.RflySimIntro\2.AdvExps\e13.WinWSL2-GPU\Readme.pdf` 来下载WinWSL2-GPU的外挂WSL2镜像（约50G），并部署到平台。安装方法简要总结如下：

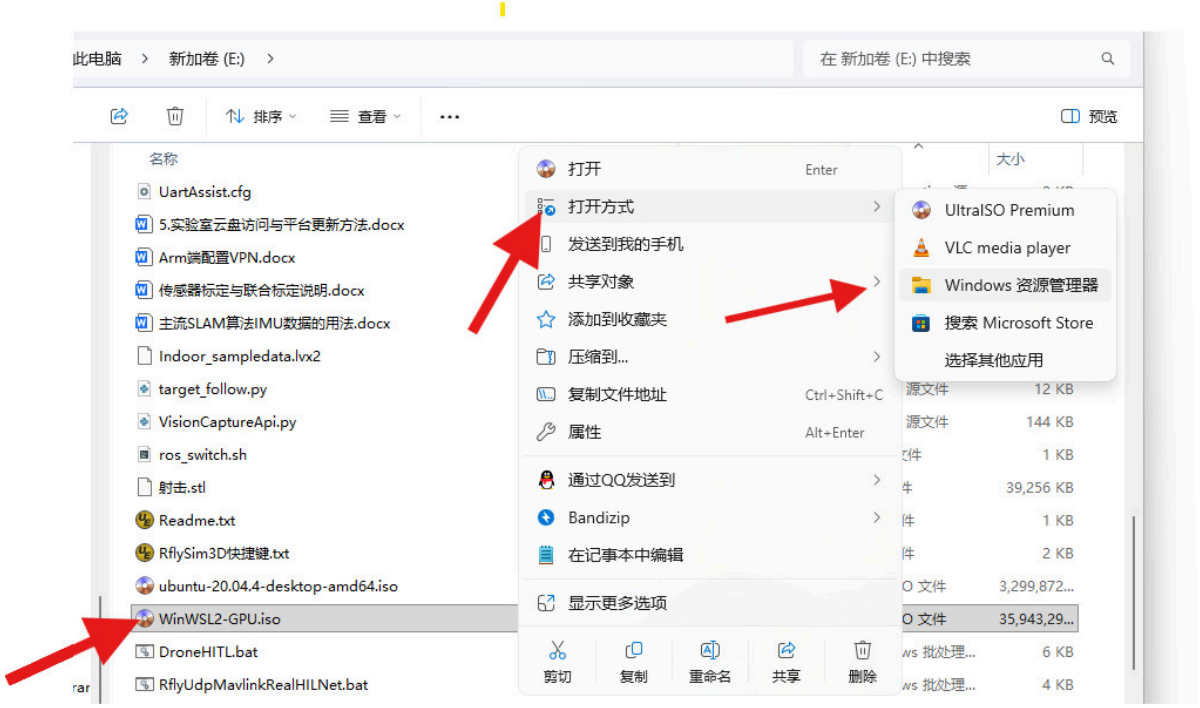
请使用百度云盘下载如下WSL2-GPU环境增量包，得到`WinWSL2-GPU-****.iso`镜像文件，大概50G左右，请耐心下载。下载链接：
https://pan.baidu.com/s/1-IdhF-GCVS9jPh4eOmas3w?pwd=suj6



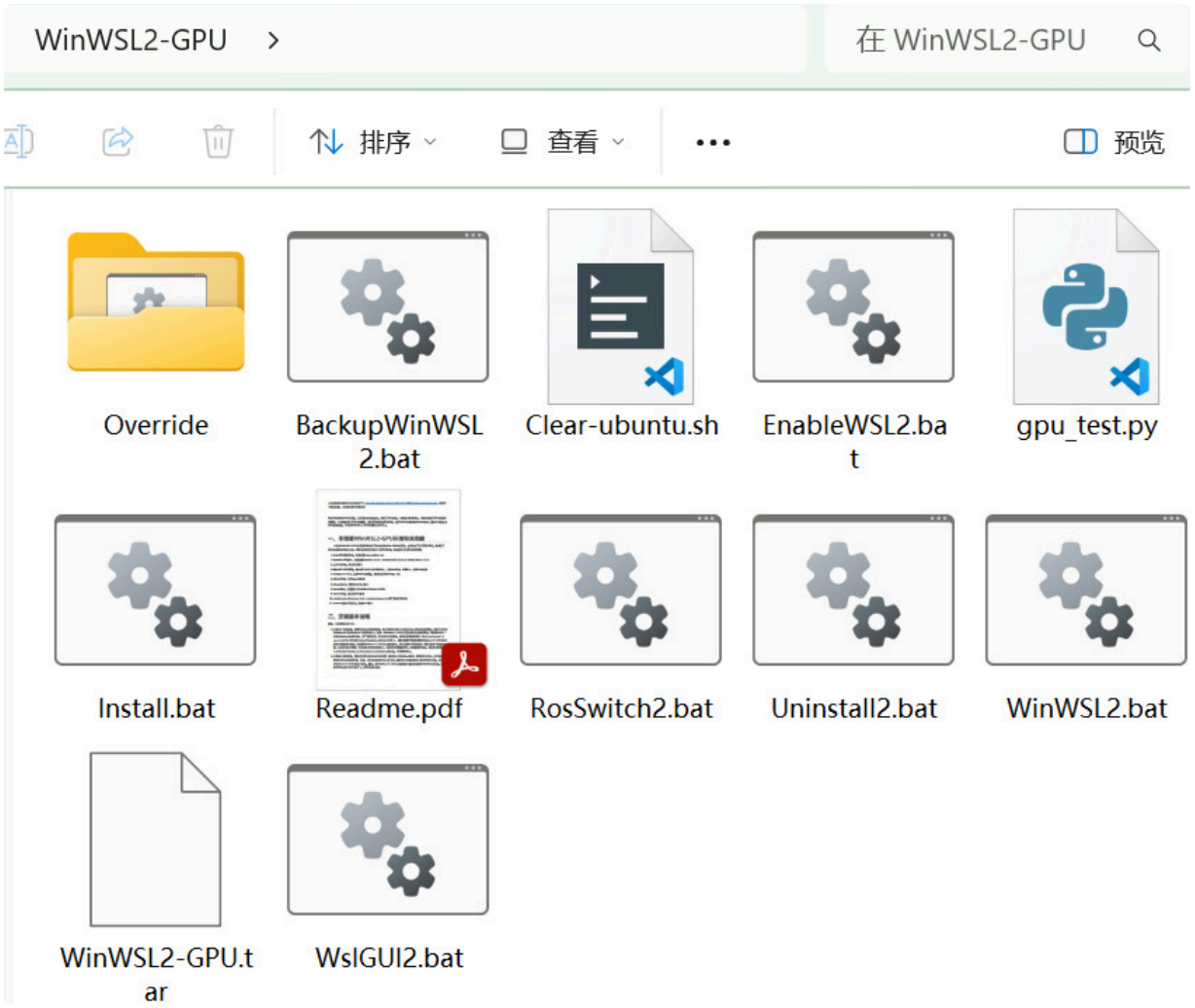
1.2 加载镜像并安装

需要加载WinWSL2-GUP.iso镜像内的内容

- **步骤一：使用windows 自带的资源管理器加载iso镜像文件**
  首先鼠标指向WinWSL2-GPU.iso，然后按鼠标右键选择“打开方式”，再然后选择“Windows 资源管理器”

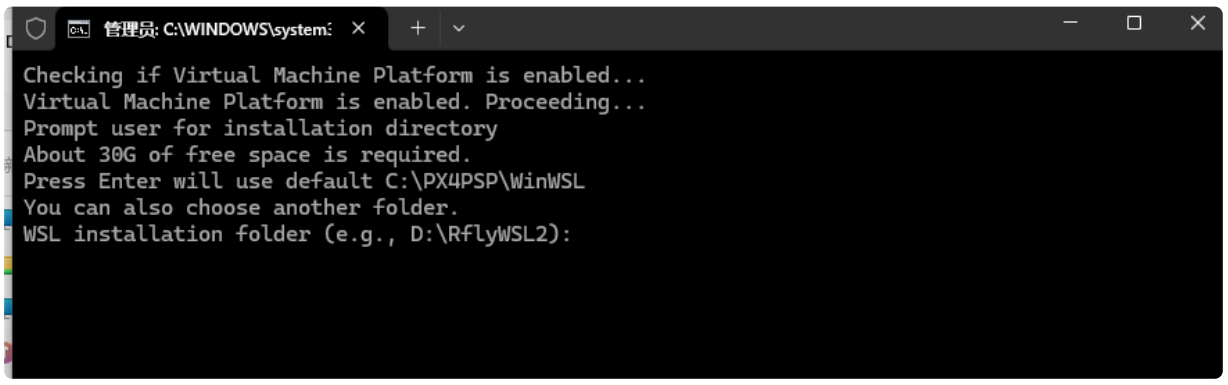


打开后，内容如下（可能不同时期的版本，文件内容略有不同。但是最基本的问都是有）



-

- **步骤二： 运行安装脚本**
  双击“Install.bat”脚本开始安装，安装过程中需要有几个地方需要注意。
  选择安装路径，因为环境镜像比较大，如果使用默认安装路径，可能会因为磁盘空间不够导致崩溃

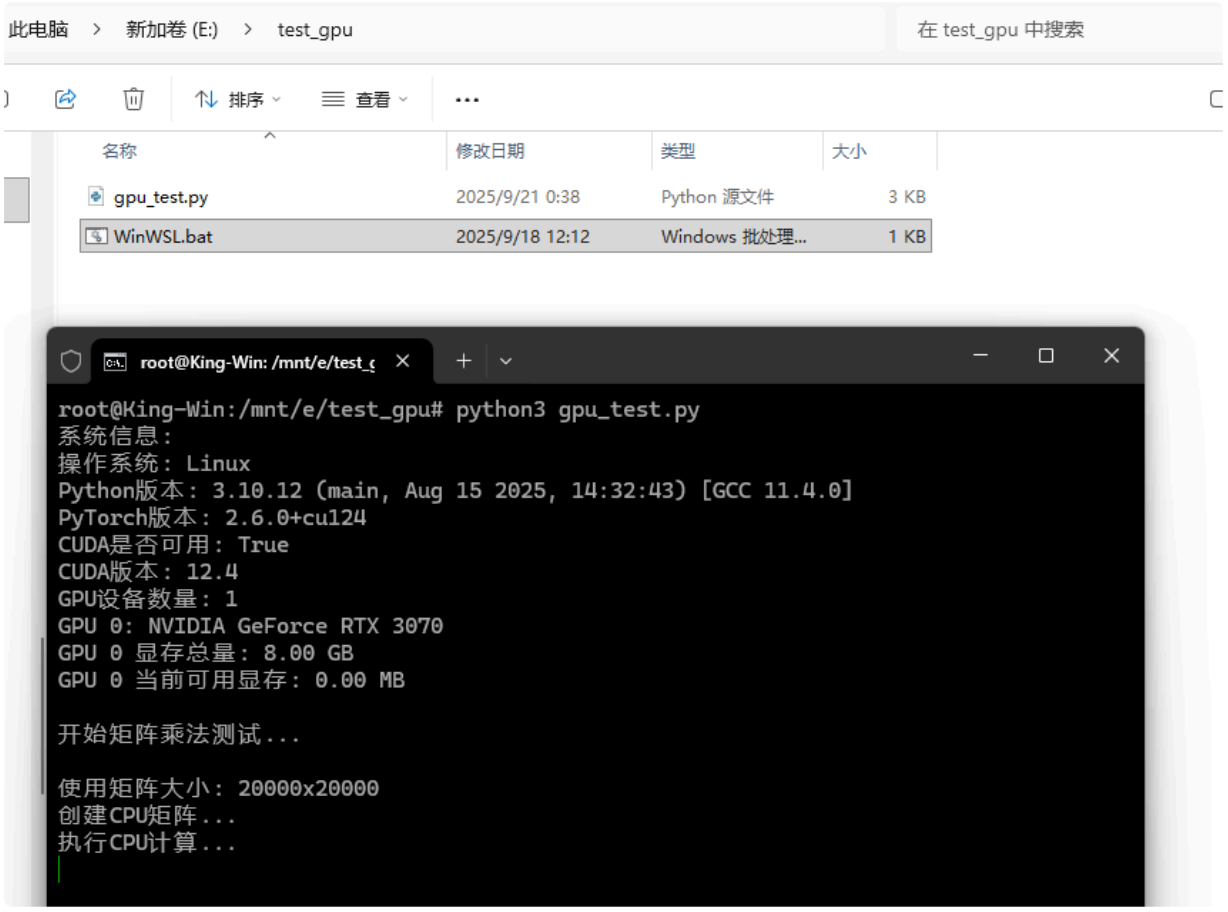


然后选择安装方式，若之前安装过外挂环境，这里会提示是否覆盖默认的WSL2环境，选择“Y”，等待精度条百分比完成即可。

```
成功：指定的值已得到保存。
WSL_PATH environment variable set successfully.
Check if RflySim-20.04 distro exists
Checking for existing RflySim-20.04 distro...
Existing RflySim-20.04 distro found. It will be uninstalled.
Confirm uninstall? (Y/N) [Y,N]?Y
Uninstalling old RflySim-20.04...
正在注销。
操作成功完成。
```

注意：使用最新版Windows11系统的用户，直接进行后续操作即可。对于Windows 11 22H2之前的系统，需要使用NAT网络运行本实验，请先**手动删除`C:\Users\<用户名>\.wslconfig`文件**，否则可能运行失败。

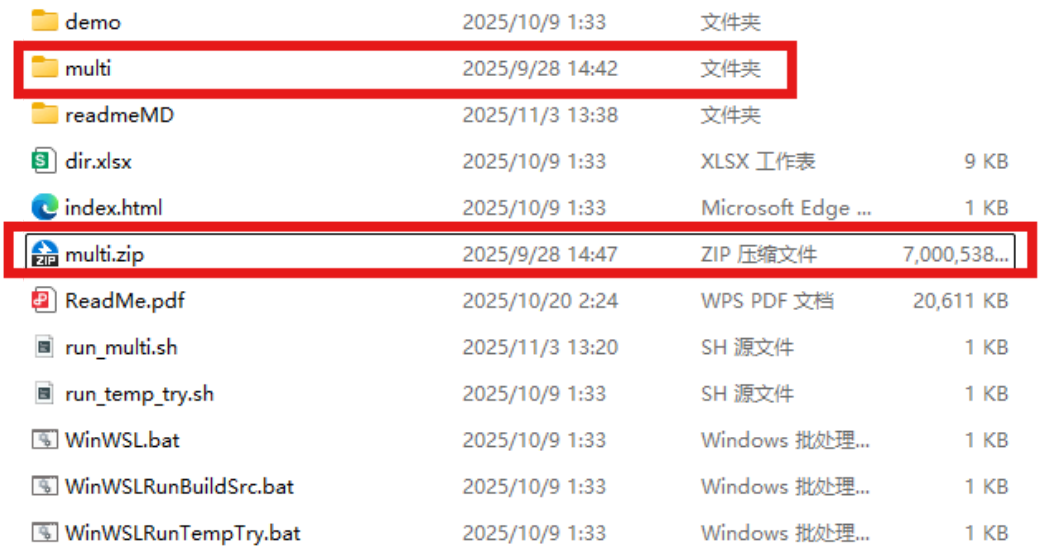
- **步骤三：运行python脚本，进行环境测试**
  把gpu_test.py 和 WinWSL.bat拷贝出来(驱动启动执行脚本，找不到挂在盘的路径)，拷贝到的文件可以随意，双击WinWSL2.bat脚本，然后输入python3 gpu_test.py，特别提醒：如果是安装后第一次打开终端，需要一定时间)



1.3 准备大模型

下载大模型以及相关的python启动文件压缩包multi.zip到当前目录下，然后解压（请自行安装解压工具，使用"解压到当前文件夹"选项，或打开WinWSL2.bat在其中输

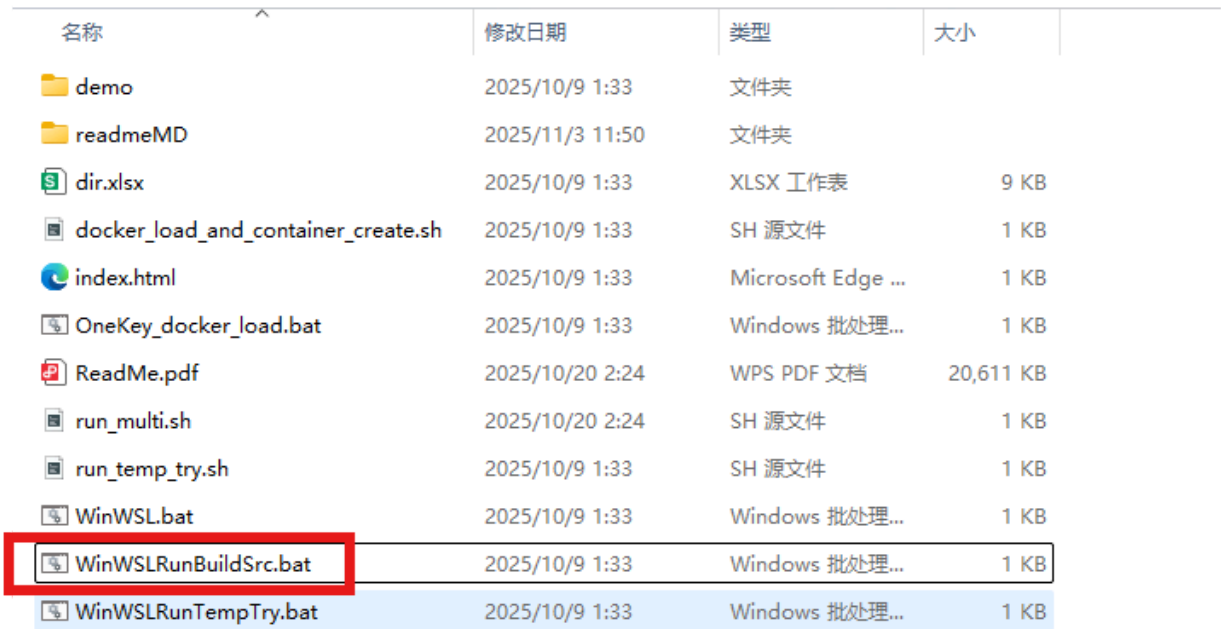
入`unzip multi.zip`），multi.zip的下载地址：
https://pan.baidu.com/s/15DXwSOOLMnKKhsaKh_1M5A?pwd=2dzq



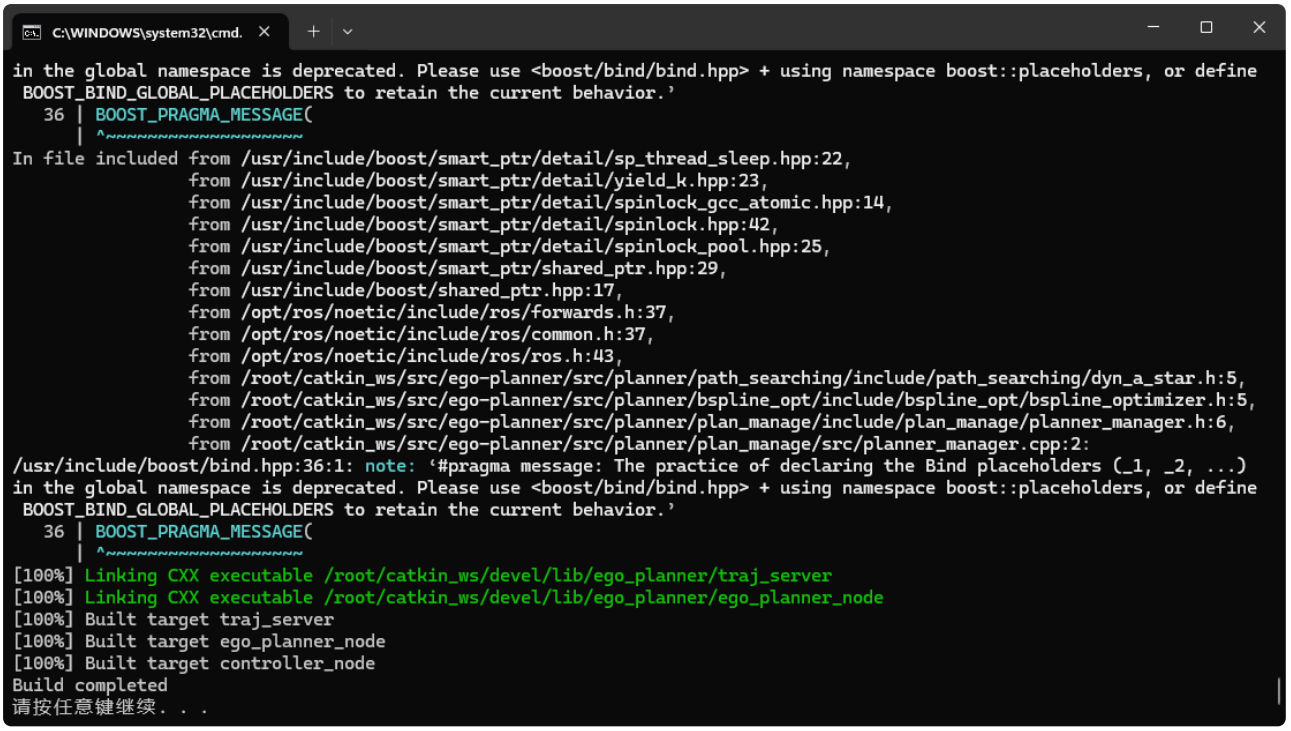
2.运行比赛程序

2.1 编译程序

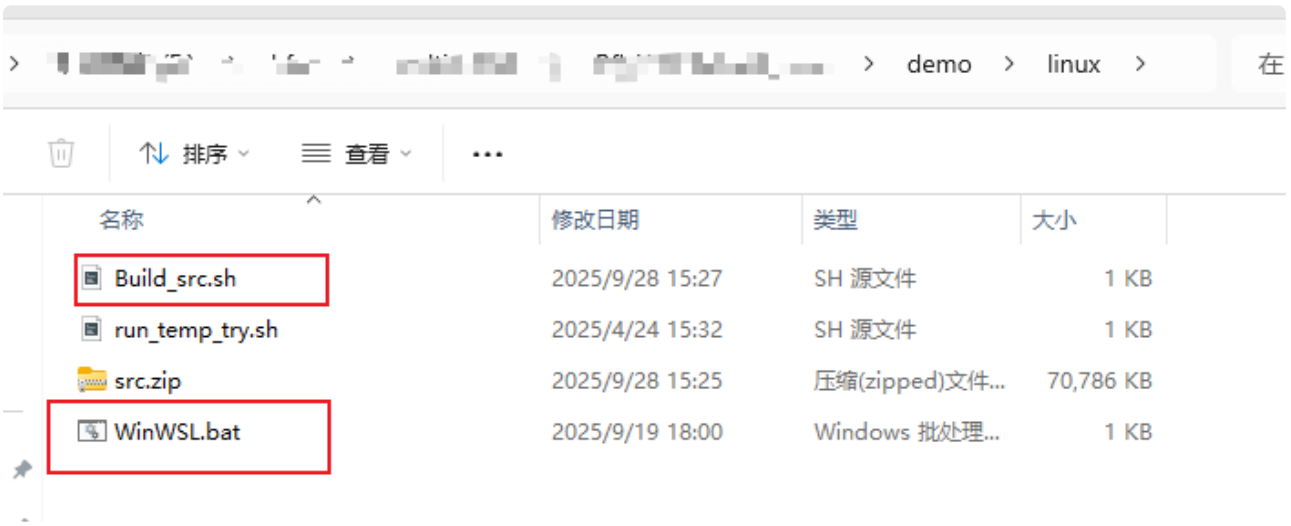
- **一键自动编译：**双击`5.LLMUavComp\WinWSLRunBuildSrc2.bat`一键运行自动编译、



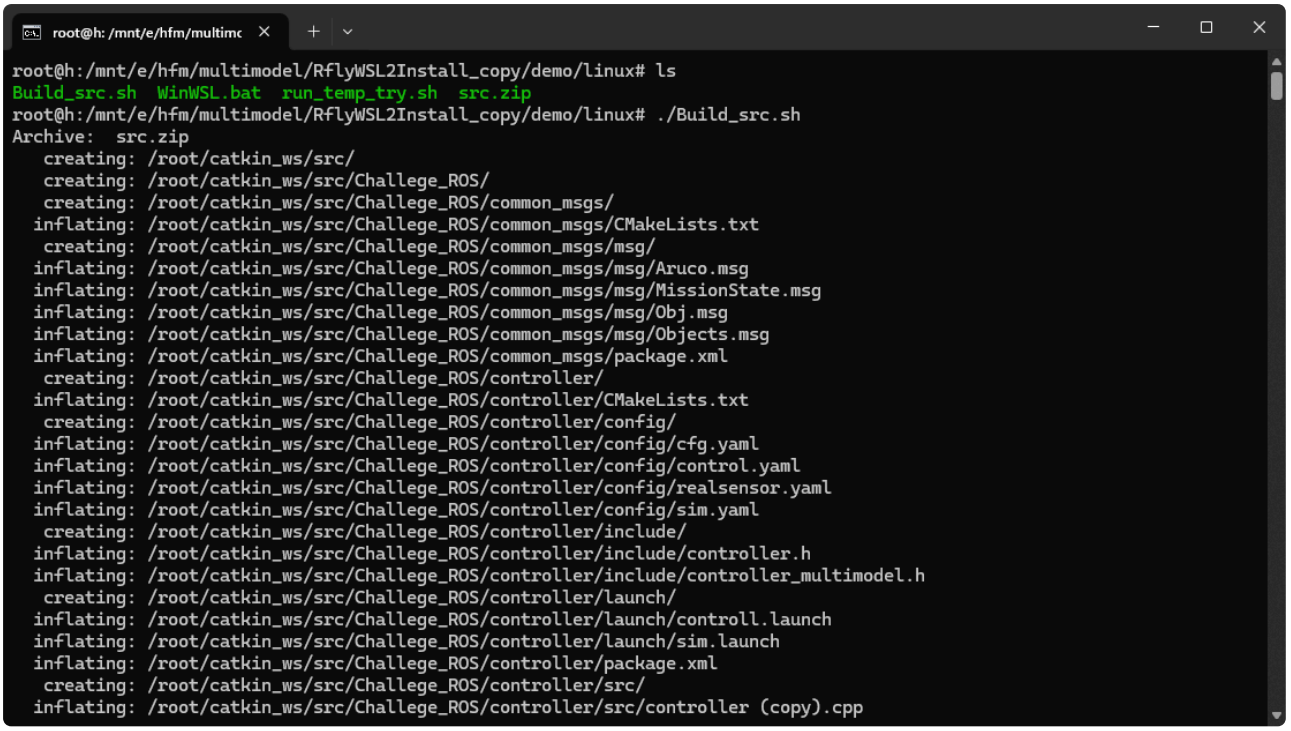
编译完成如下图：



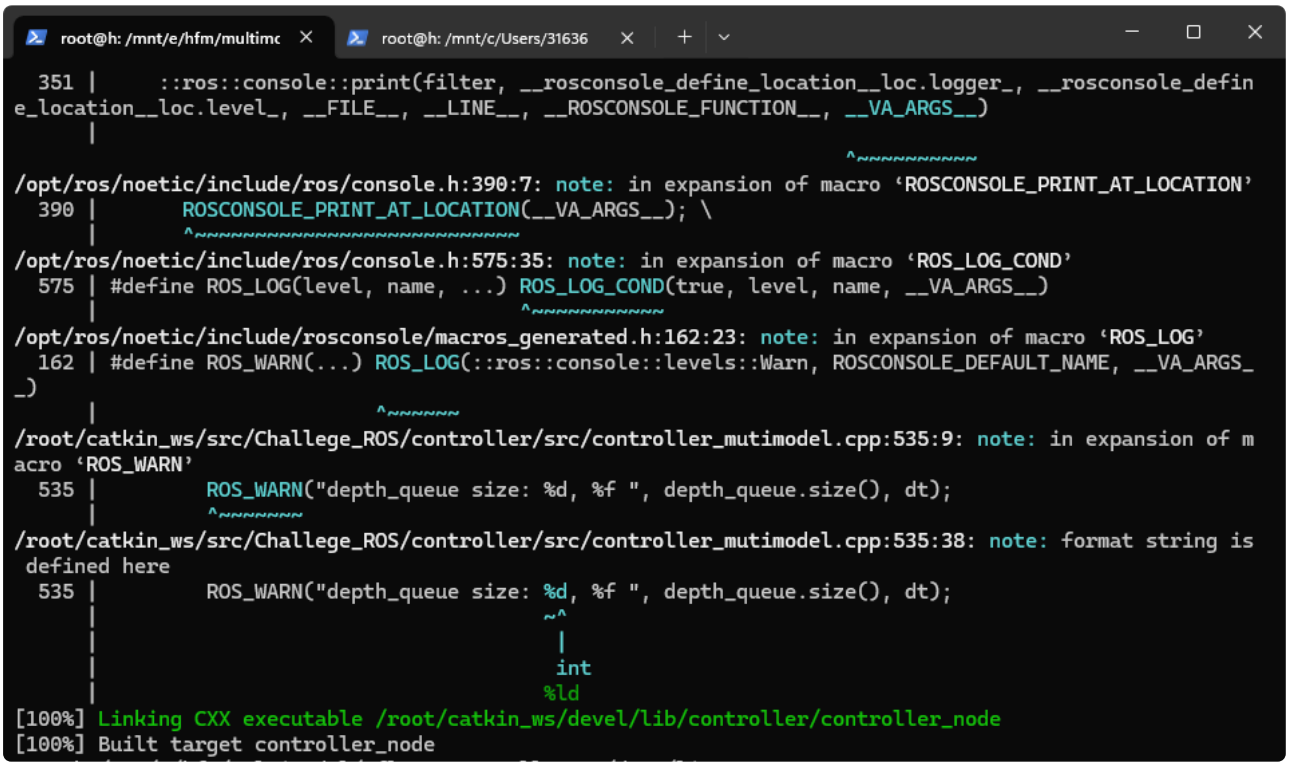
- **手动编译方案，仅作为选做项目：**进入demo/linux文件夹



双击WinWSL2.bat打开wsl2环境，输入`./Build_src.sh`，一键编译程序

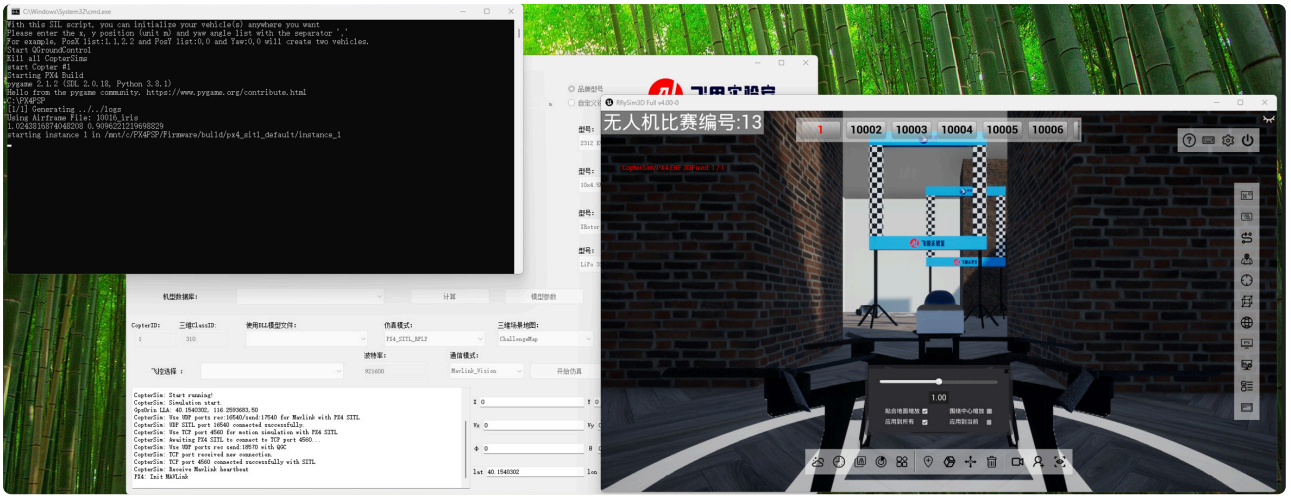


编译成功如下图所示：

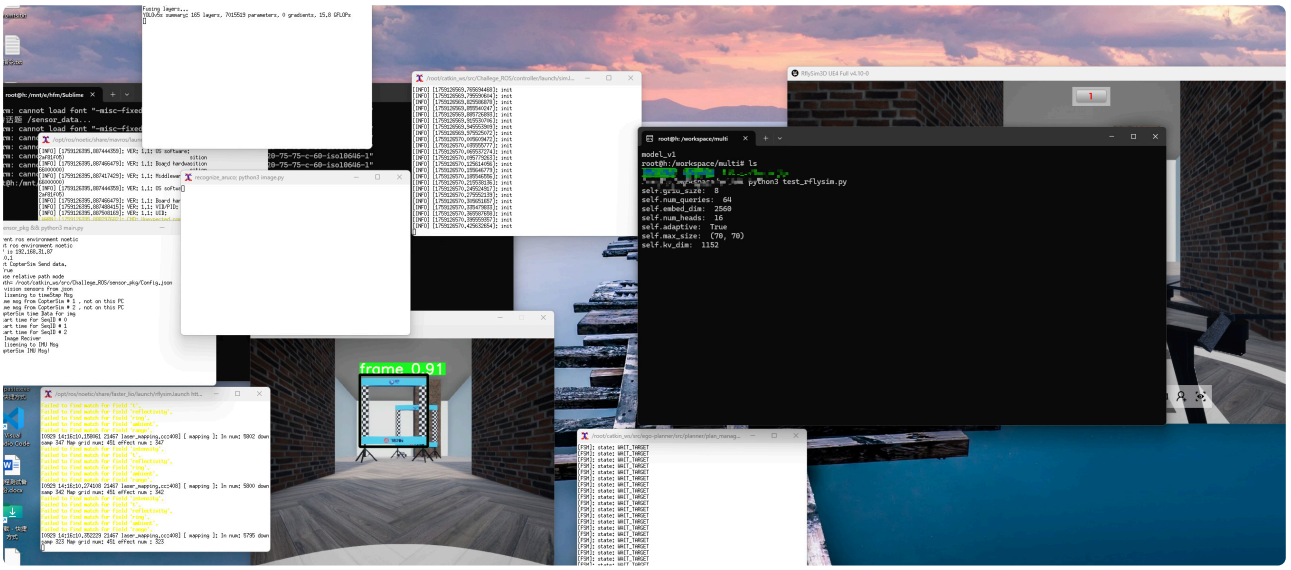


2.3 运行仿真环境

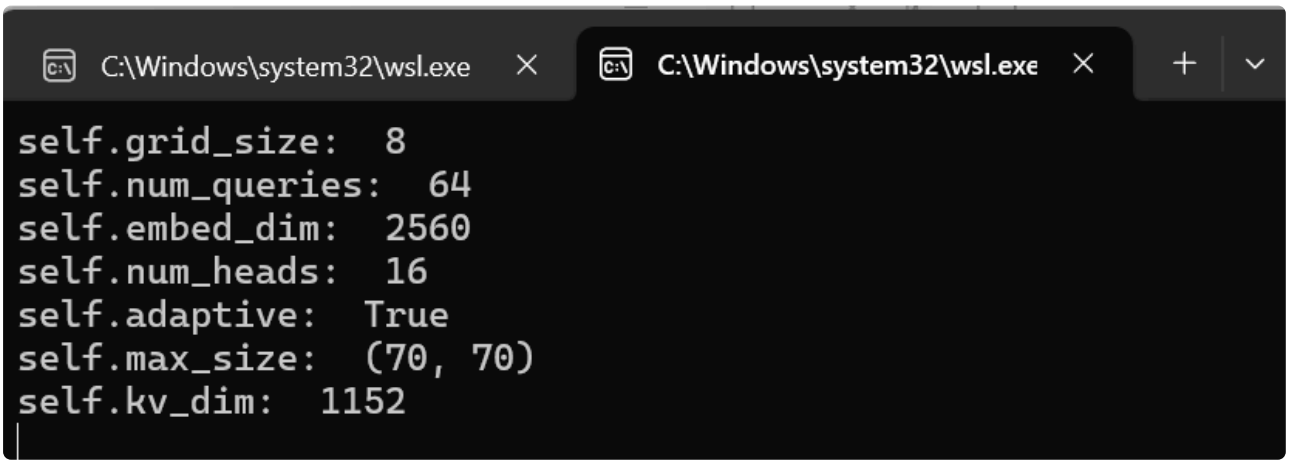
- **步骤一：**双击`demo\win`下的SITLPosStrOnekey.bat, 等待初始化完成(Rflysim3D出现`CopterSim/PX4 EKF 3DFixed: 1/1`)。



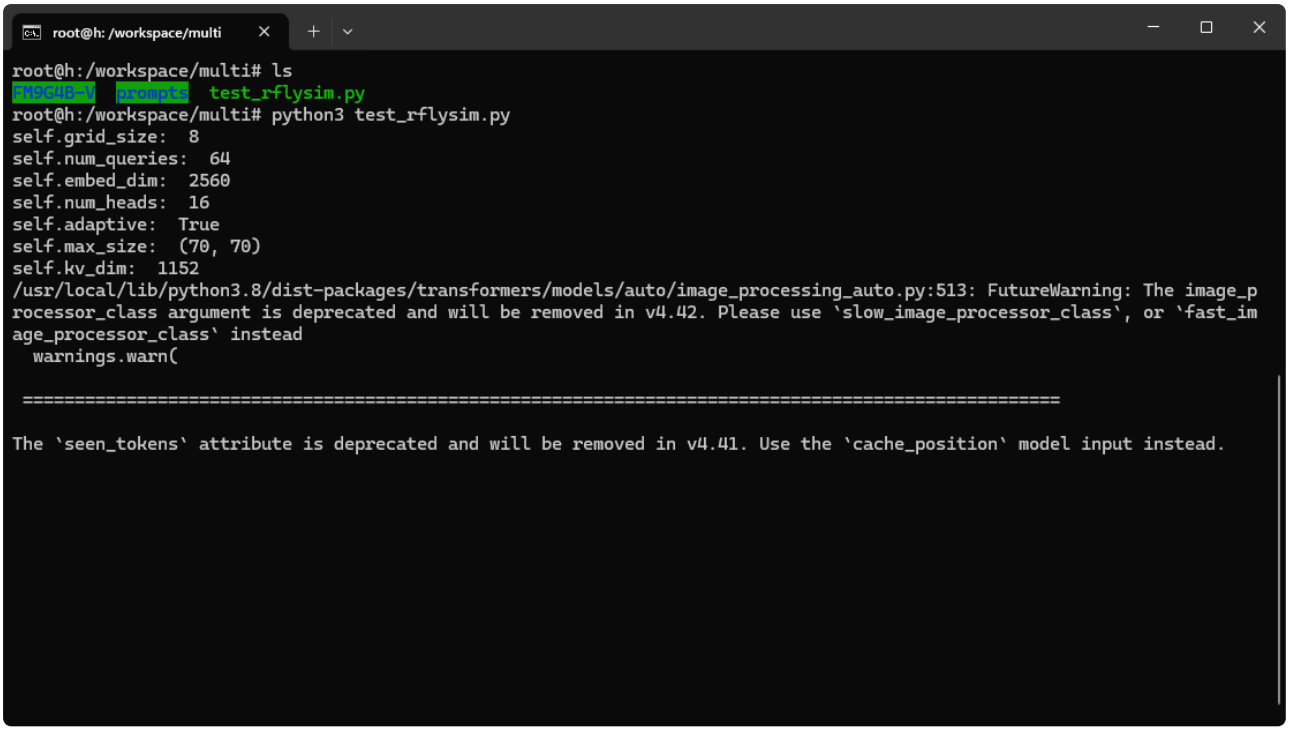
- **步骤二：**双击`5.LLMUavComp\WinWSLRunTempTry2.bat`脚本，一键运行比赛例程。



在此时能找到如下页面，需要等待大模型加载结束再输入指令。

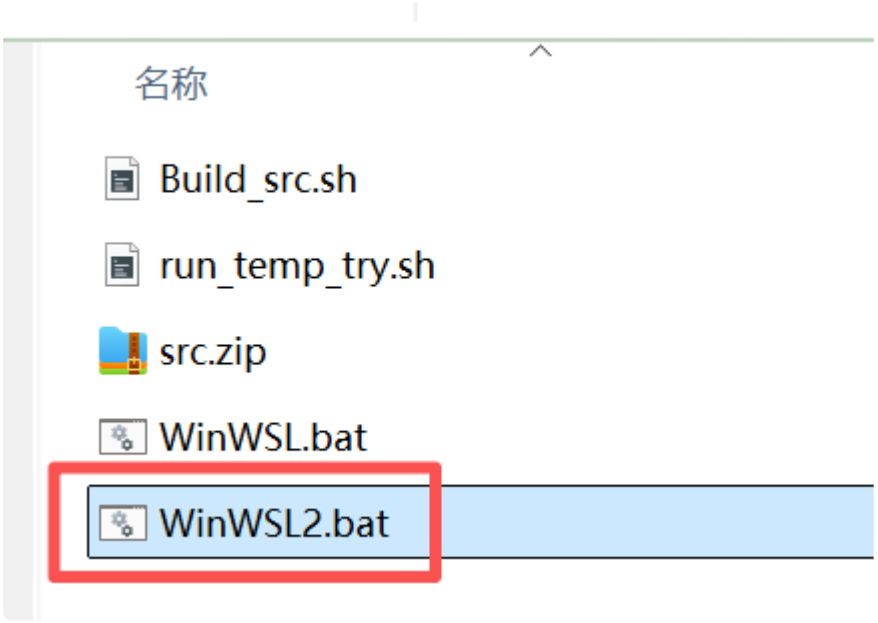


注意：“self.kv_dim: 1152”语句处要等待一段时间，才能让大模型完全加载完，请耐心等待。大模型加载速度跟电脑性能有关，若没显示输入框说明大模型还在加载。

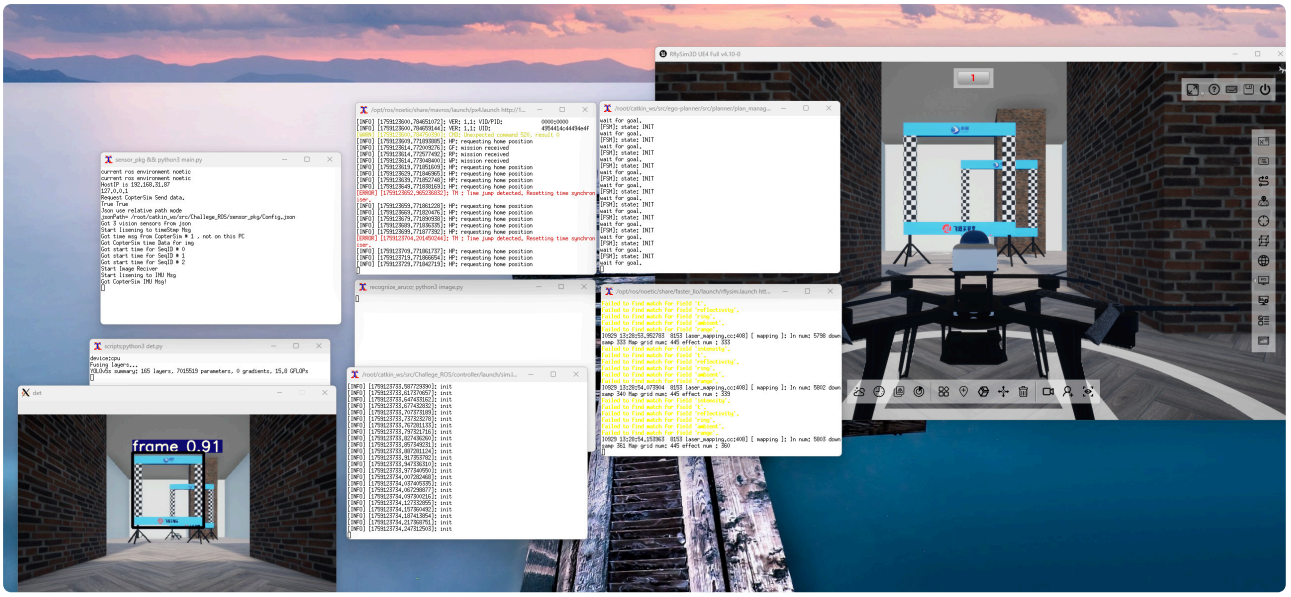


注意：**若WinWSLRunTempTry2.bat自动执行失败**，可使用如下手动步骤

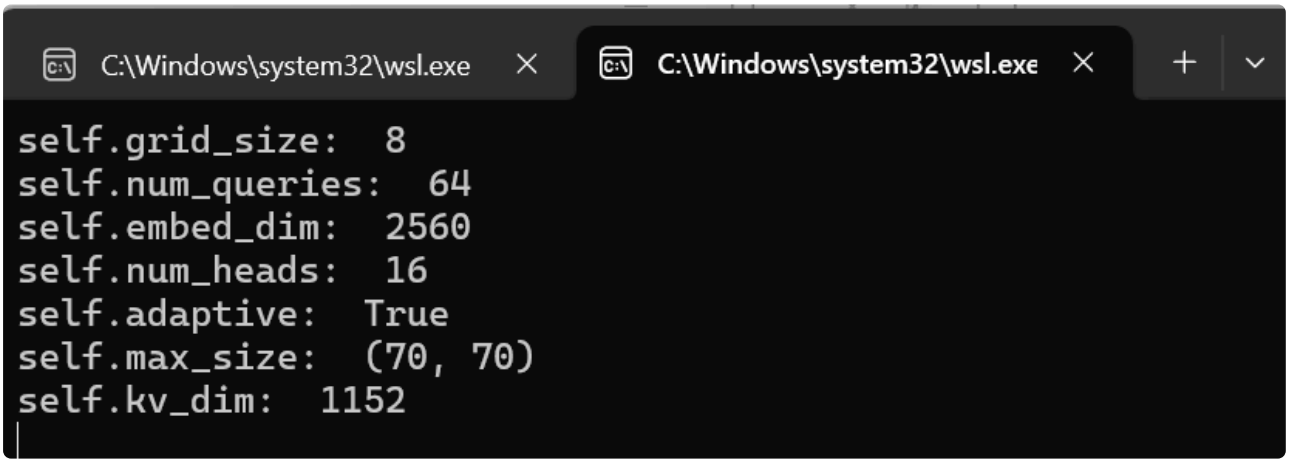
- **选做步骤，手动程序（和步骤二，二选一即可）：**进入demo/linux文件夹，双击WinWSL2.bat打开一个WSL2环境



然后再，输入`./run_temp_try2.sh`等待窗口加载完毕：



再双击一个WinWSL2.bat，输入`cd multi`进入大模型multi目录，然后输入`ls`命令，确认test_rflysim.py等文件是否如下图存在当前目录，且为绿色（可执行）。然后，输入`python3 test_rflysim.py`运行大模型，并等待大模型加载结束再输入指令。注意：“self.kv_dim: 1152”语句处要等待一段时间，才能让大模型完全加载完，请耐心等待。大模型加载速度跟电脑性能有关，若没显示输入框说明大模型还在加载。在此时能找到如下页面，需要等待大模型加载结束再输入指令。

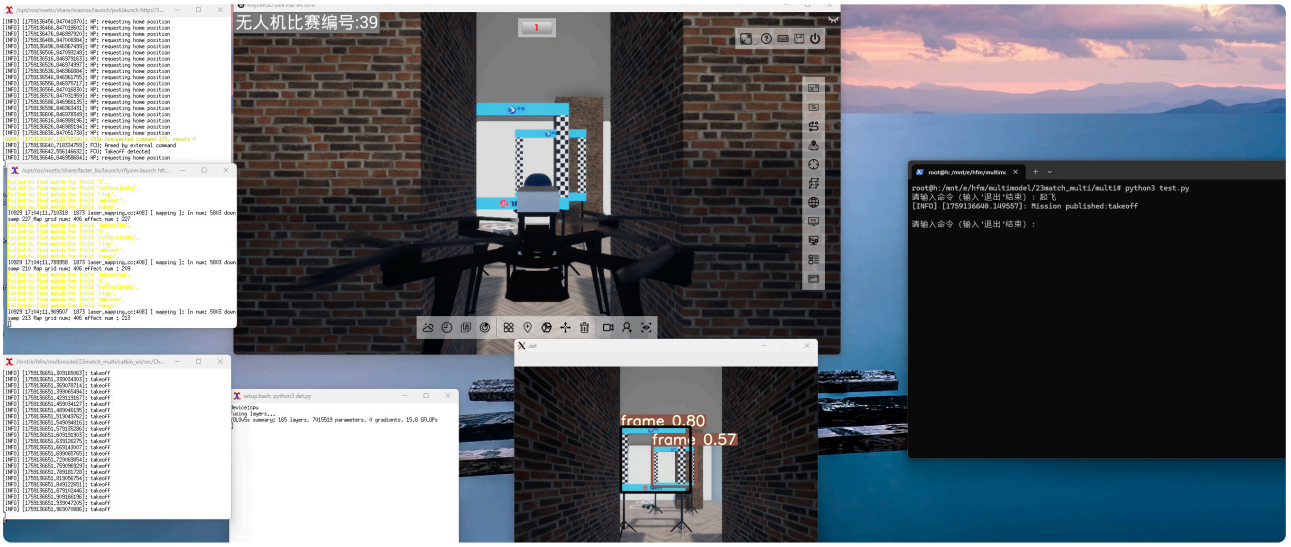


大模型加载完毕会有输入框，如下图所示：

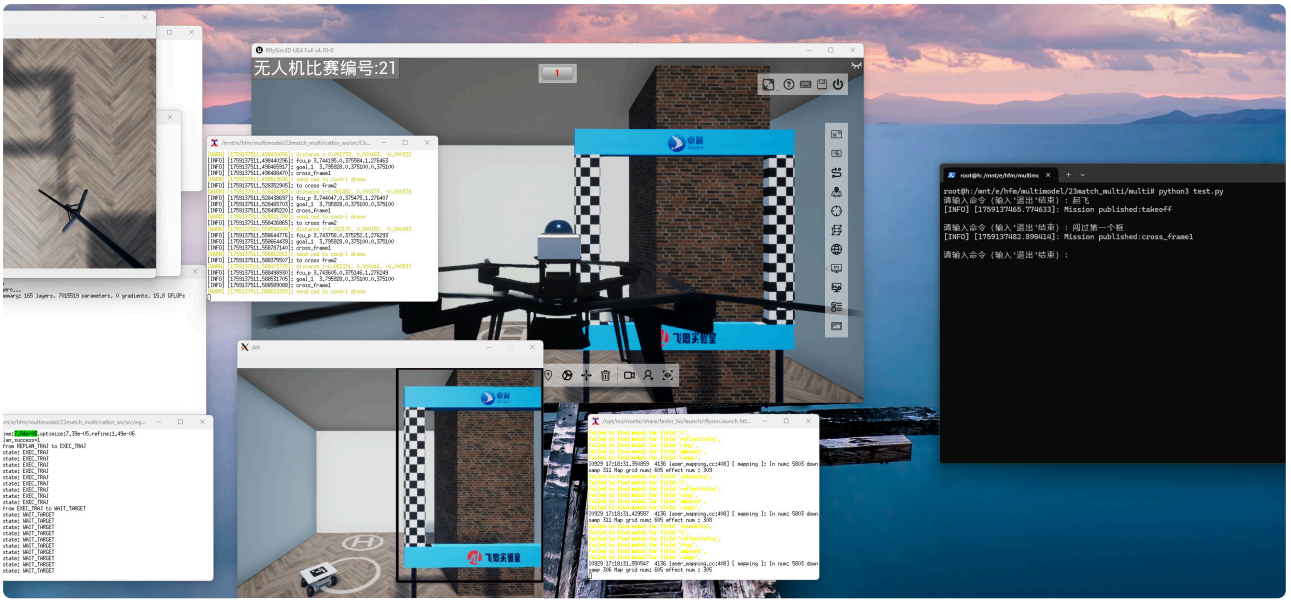
```
l# cd demo/linux/
linux# python3 test_rflysim.py
self.grid_size:  8
self.num_queries:  64
self.embed_dim:  2560
self.num_heads:  16
self.adaptive:  True
self.max_size:  (70, 70)
self.kv_dim:  1152
/usr/local/lib/python3.8/dist-packages/transformers/models/auto/image_processing_auto.py:513: FutureWarning: The image_processor_class argument is deprecated and will be removed in v4.42. Please use `slow_image_processor_class`, or `fast_image_processor_class` instead
  warnings.warn(

 ================================================================================================

The `seen_tokens` attribute is deprecated and will be removed in v4.41. Use the `cache_position` model input instead.
您好，我是由启元实验室主导研发的九格通用基础大模型。
我在海量的文献中学习了自然语言的各种细节。我对中英文最为熟练，也对其他语言有一定了解。随时欢迎您的提问，但在我未知的领域，我提供的回答可能不够准确。请注意，我的答案只是一个参考。我也具备图片理解能力，您还可以上传图片来询问。请输入命令（输入'退出'结束）：起飞
```

2.3 实验效果

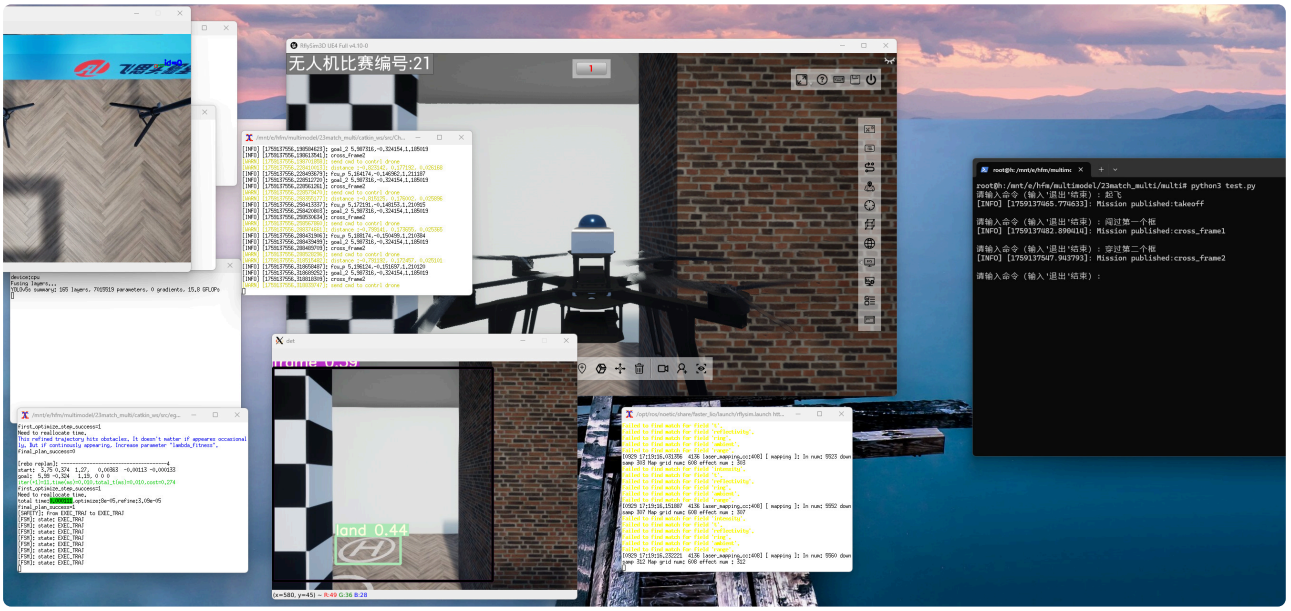
起飞无人机。 例如，在下图的右边窗口，输入“起飞”，可以看到飞机自动起飞。



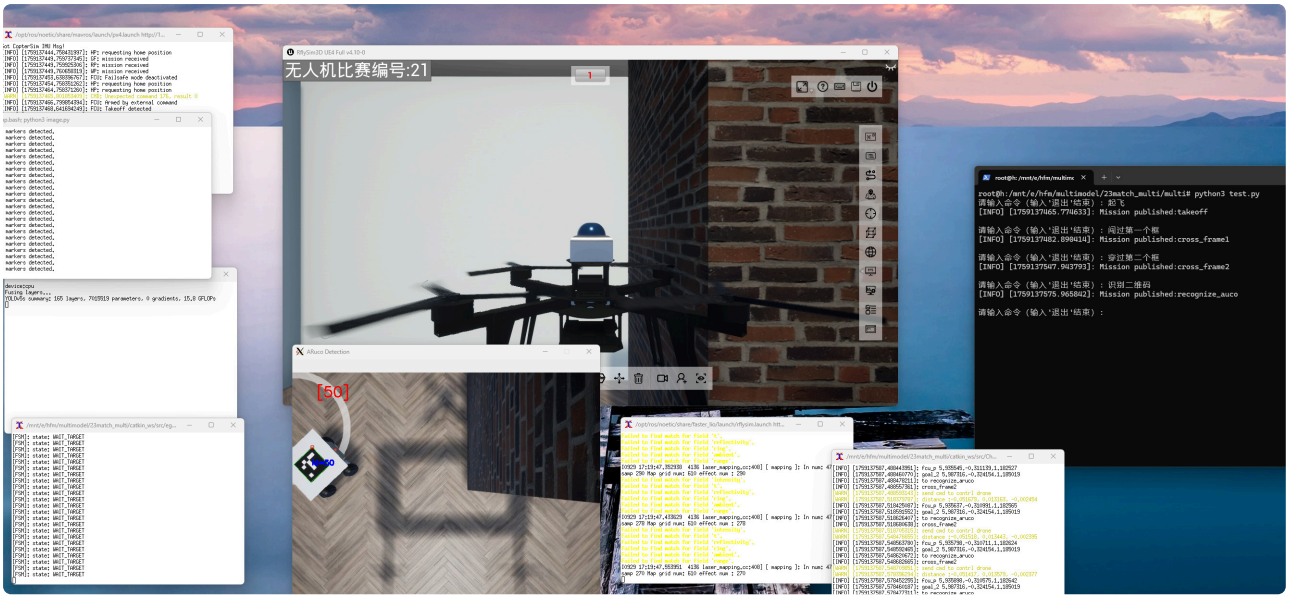
穿过框一。例如，在下图的右边窗口，输入“闯过第1个框”，可以看到飞机自动穿越第一个框。



穿过框二。例如，在下图的右边窗口，输入“穿过第2个框”，可以看到飞机自动穿越第二个环。注：闯过和穿过语义一样，理论上大模型是能够区分的（部分情况下可能失效，请调整提示词）。



识别二维码。例如，在下图的右边窗口，输入“识别二维码”，可以看到飞机自动读取二维码的值。



3.开发说明

无人机控制和ROS取图之类接口可以学习例程`8.RflySimVision\1.BasicExps\5_CompSlamNav\Readme.pdf`，这里提供大模型相关的开发。

大模型运行文件位于`5.LLMUavComp\multi\test_rflysim.py`，提示词文件位于`5.LLMUavComp\multi\prompts\basic_cn.txt`。你可以直接在windows下直接对`5.LLMUavComp\multi`里面的文件进行修改，大模型主要可以修改提示词让他输出你想要的结果，然后通过在`test_rflysim.py`你需要对大模型输出的内容进行处理，提取出关键结果，用关键结果来做任务。如本例中：

提示词：

````
你是一名助手，正在帮助我使用定义好的指令来帮我控制无人机的飞行完成我指定的一系列任务。
以下是您可以用来控制无人机的一些命令。

takeoff  --指令解释：该指令用于启动无人机的垂直升空程序，使无人机从地面或起降平台平稳上升至预设的悬停
liftoff --指令解释：提升飞行高度。
flight --指令解释：激活飞行器的空中运行状态准备程序。
pass_frame1 --指令解释：引导设备从边界外围区域穿过首个视觉设定边界。
cross_frame1 - -指令解释：引导设备从起始框型结构中心穿越。
traverse_boundary - -指令解释：驱动航行单元越过首阶段的构造边缘。
cross_structure1 --指令解释： 引导设备穿越初始环型结构。
cross_frame2 --指令解释： 控制设备通过次级目标设定好的框形通道。
pass_frame2 --指令解释： 控制设备通过后续单元目标设定边界。
cross_structure2 --指令解释： 引导设备穿越下一阶段目标环型结构。
detect_marker --指令解释：触发图像识别模块以探测环境中的标记符号。
recognize_aruco --指令解释：激活视觉识别模块以检测环境中的编码图案。
analyze_visual_code --指令解释：对视觉图案中蕴含的编码信息进行解析。
recognize_H --指令解释： 启用目标检测功能以定位着陆参考标识。
recognize_land --指令解释：启用目标检测功能对地面参考标识进行识别。
recognize_locate --指令解释：对场景中用于定位的形状图元进行识别。
detect_surface_marker --指令解释: 启动下视识别机制以扫描着陆辅助符号。
land --指令解释： 执行地面接触协议，停止空中阶段运行。
hover --指令解释：保持空中设备在当前位置悬停。
cross_line --指令解释：指引装置穿过一条直线标记。
recognize_pattern --指令解释：启动模式识别系统以分析周围环境中的复杂图形结构。
ascend --指令解释：调整飞行高度至指定水平。
landing --指令解释： 进入着陆准备状态。

当我要求你做某事时，按以下结构回答只能一次完成不能重复：
{1. 解释思路
以简明语言说明解决思路和步骤
2. 控制命令
```command（要执行的command放在这里，不允许分成一段一段或者在其他地方。）
```}
````

大模型的输出会根据提示词最下面要求格式进行输出。

```python
# 解析指令
    def extract_code(self, content):
        """
        Extracts the command from a response.
        :param content:
        :return:
        """
        code_block_regex = re.compile(r"```(.*?)```", re.DOTALL)
        code_blocks = code_block_regex.findall(content[0])
        if code_blocks:
            full_code = "\n".join(code_blocks)
            if full_code.startswith("command"):
                full_code = full_code[8:]

            return full_code.strip()
        else:
            return None
```

这里代码进行一个正则匹配提取出指令，指令然后被ros话题`/mission_command`发布出去。

```python
    rospy.init_node('mission_commander')
    pub = rospy.Publisher('/mission_command', String, queue_size=10)
    multi_rflysim = Multi_RflySim()
    while True:
        command = input("请输入命令（输入'退出'结束）：")
        if command == "退出":
            break
        mission = multi_rflysim.process(command)
        print("mission:", mission)
        if mission:
            pub.publish(mission)  # 发送任务指令
            rospy.loginfo("Mission published")
```

与大模型交互的控制端程序位于`/root/catkin_ws/src/Challege_ROS/controller/src/controller_mutimodel.cpp`

它会订阅`/mission_command`话题，拿到其中的控制指令，然后去进行接口匹配，完成对应的任务，具体实现方式自行阅读代码。

若想进行本例程外的其他任务，需要自行在`controller_mutimodel.cpp`进行接口的编写，并在提示词进行接口提示。