本地版大模型部署及调用实验

1. 实验目的

通过 WinWSL2-GPU 的 WSL 同构环境在本地离线部署 ollama，完成对轻量化大语言模型的推理调用，充分利用宿主机 GPU 算力，同时在仿真链路上（SITL/HITL）打通无人机任务控制接口，展示本地大模型推理的端到端实战流程。

2. 实验要求

- 软件要求：Windows 10及以上版本；RflySim工具链[1]；MATLAB2022B以上版本。
- 硬件要求：笔记本/台式电脑1台[2]。

3.实验地址

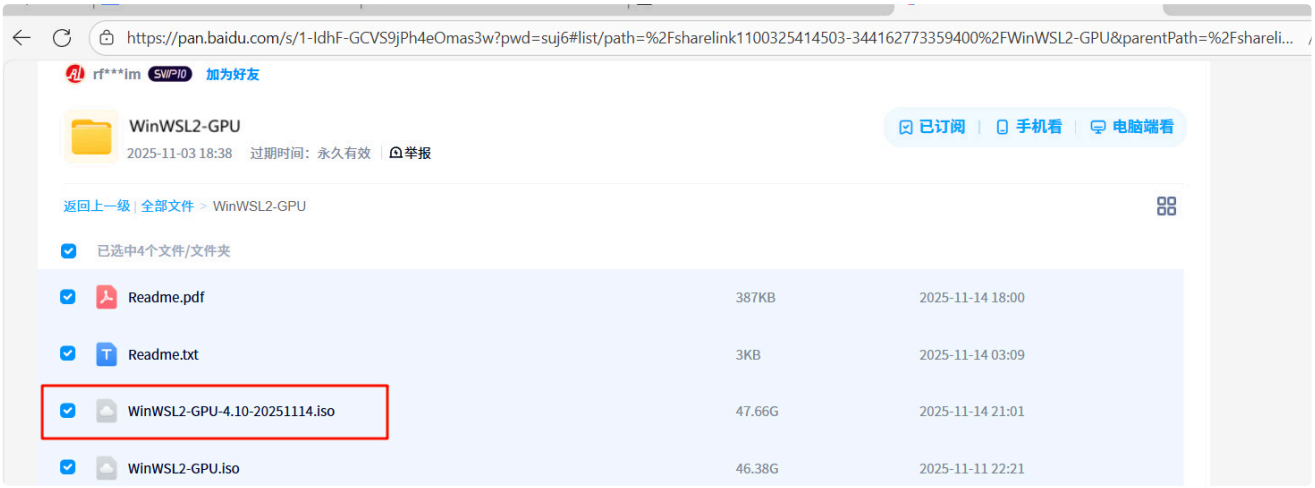
例程目录：[安装目录]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e14.LocalLLMDepUse

- ./LLM_UAV_init.m：MATLAB 脚本进行参数初始化
- ./Main_OpenAI_api.py：大模型服务端程序，调用 OpenAI API 并输出标准化控制命令
- ./RflyUdpMavlinkRealSim.bat：启动 PX4 飞控在环仿真的批处理文件
- ./index.html：Web界面文件

4. 实验内容或步骤

4.1 步骤1：WSL2-GPU增量环境包下载

本实验中需要使用WinWSL2-GPU的外挂WSL2镜像（约50G），并部署到电脑中。请使用百度云盘下载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



后续步骤请务必按照`[RflySim安装路径]\1.RflySimIntro\2.AdvExps\e13.WinWSL2-GPU\Readme.pdf`实验来进行。确保安装完成。

4.2 步骤2：镜像运行及测试

双击运行 WinWSL2.bat，在打开的终端中输入：`ollama list`，可以看到该镜像中已安装`qwen3:0.6b`模型。

```
root@[redacted]:/mnt/c/Users/biyan# ollama list
NAME            ID              SIZE      MODIFIED
qwen3:0.6b      7df6b6e09427    522 MB    3 weeks ago
```

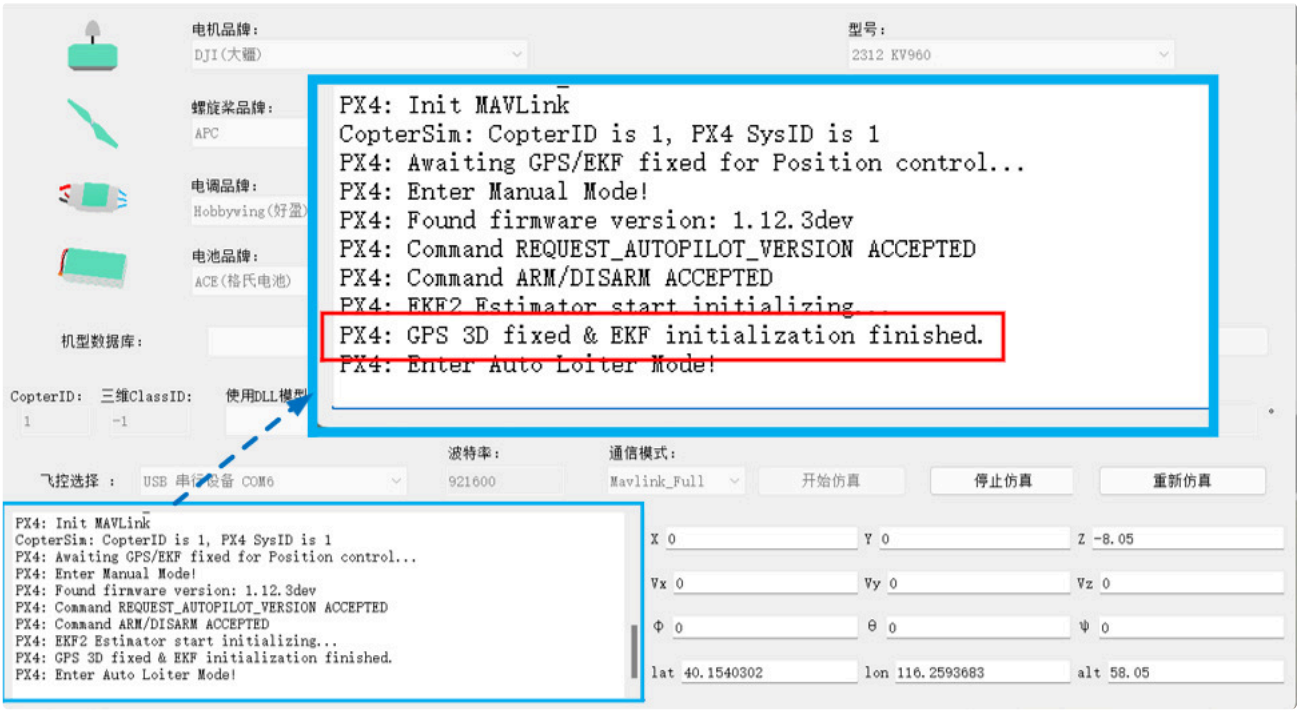
可以运行`ollama run qwen3:0.6b`启动qwen模型，并输入文字即可对话。

```
root@Byzeal:/mnt/c/Users/biyan# ollama run qwen3:0.6b
>>> 你好
Thinking...
好的，用户发来“你好”，我需要回应。首先，用户打招呼，这很友好，但可能只是简单的问候。我应该保持自然，不显得生硬。可以回应
“你好！有什么可以帮助你的吗？”这样既回应了问候，又询问了需求，同时保持了轻松的氛围。不需要太多解释，让用户感到被重视。此外
，用户可能希望得到进一步的帮助，所以询问是否需要帮助是合适的。整个回应要简洁自然，符合对话的流畅性。
...done thinking.

你好！有什么可以帮助你的吗？
```

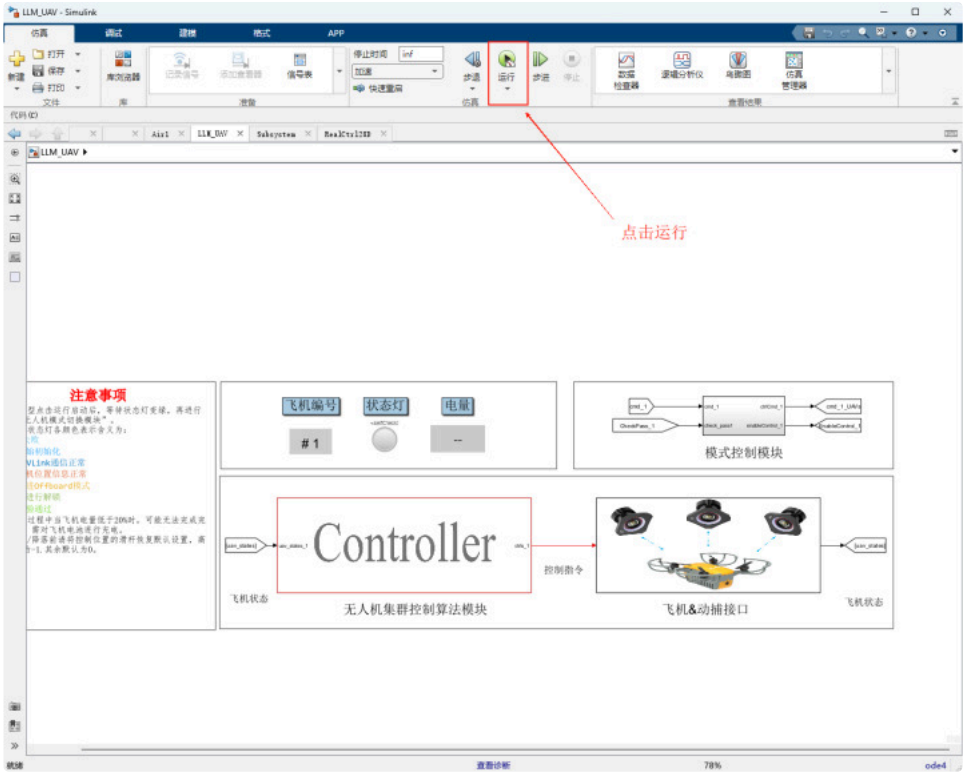
4.3 步骤3：启动飞控软件在环仿真

在本实验所在的文件夹中，双击运行批处理文件`RflyUdpMavlinkRealSim.bat`，启动 PX4 飞控软件在环仿真环境。等待仿真环境初始化完成。脚本将会启动 1 个 QGC 地面站， 1 个 CopterSim 、1 个 RflySim3D 软件， 等待CopterSim软件下侧日志栏必须打印出`GPS 3D fixed & EKF initialization finished`字样代表初始化完成。如下图所示：



4.4 步骤4：运行外部控制程序

在同一文件夹下打开`LLM_UAV.slx`模型文件，在 MATLAB/Simulink 中点击「运行」，启动模型执行。



4.5 步骤5：启动文本大模型服务端并发送指令

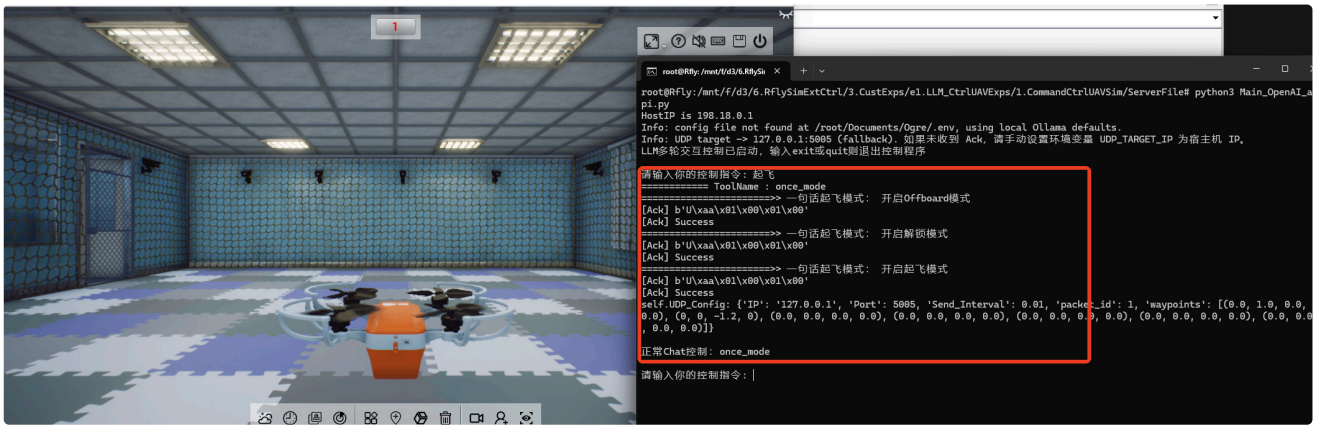
使用已配置好 Python 环境的 Visual Studio Code 打开 `\ServerFile\Main_OpenAI_api.py` 程序，并运行该脚本以启动大模型服务端。

双击 winwsl2.bat 启动WSL2终端，执行如下命令

```
python3 Main_OpenAI_api.py
```

随后在终端中输入自然语言指令，程序会自动解析语义并选择对应的控制模式。例如：

- 输入：**一键起飞。** 程序将依次启动 Offboard 模式、解锁模式和起飞模式，最终完成飞机解锁并起飞升空。
- 输入：**降落。** 程序将启动 `Land_mode` 模式，控制飞机降落至地面。
- 输入：**锁定。** 程序将启动 `Lock_mode` 模式，使飞机电机停止转动。
- 输入：**解锁。** 程序将启动 `UnLock_mode` 模式，使飞机电机解锁。



4.6 步骤6：下载本地版多模态大模型增量包（选做）

这里提供启元多模态大模型的离线增量包及对应 Python 示例。解压后会得到 `multi/` 目录，内含：

- `FM9G4B-V/`：启元多模态权重和自定义 Transformers 代码（需 trust_remote_code 加载）。
- `prompts/`：中文提示词，定义飞行动作和回复格式。
- `test_rflysim.py`：本地调用启元大模型并通过 ROS 发布指令的示例脚本。

使用步骤：

1. 下载 multi.zip（https://pan.baidu.com/s/15DXwSOOLMnKKhsaKh_1M5A?pwd=2dzq）到当前目录。
2. 解压到当前文件夹（使用解压工具或在 WinWSL2.bat 打开的 WSL 终端执行 `unzip multi.zip`）。
3. 确认 GPU/CUDA 与 PyTorch 可用，磁盘有足够空间存放 `FM9G4B-V`。
4. 启动WinWSL2.bat打开终端并进入解压后的 `multi` 目录，运行 `python3 test_rflysim.py`，按提示输入中文指令，模型会生成动作并发布到 `/mission_command` ROS 话题。
   具体视觉任务示例见 [安装目录]/RflySimAPIs/8.RflySimVision/1.BasicExps/5.LLMUavComp/Readme.pdf。

5. 关键知识点

关键知识点1：本地文本大模型调用

Main_OpenAI_api.py 默认通过 ollama 的 Python Client 调用 `qwen3:0.6b`，可用环境变量覆盖：

- `OLLAMA_HOST` / `OLLAMA_BASE_URL`：Ollama 服务地址（Python Client 用主机地址即可，兼容结尾带 /v1）。
- `OLLAMA_MODEL`：模型名，默认 `qwen3:0.6b`。

若需直接 REST 方式测试：

```python
import requests
OLLAMA_URL = "http://localhost:11434/api/generate"  # 如改端口/地址请同步修改
PROMPT = "请输入一组飞行[ x, y, z ]点并给出对应姿态，输出列表格式"
resp = requests.post(OLLAMA_URL, json={"model": "qwen3:0.6b", "prompt": PROMPT, "stream": False}, timeout=30)
resp.raise_for_status()
print(resp.json().get("response", "").strip())
```

关键知识点2：多模态本地模型文件说明

- `test_rflysim.py`：主脚本。启动 ROS 节点 `mission_commander`，发布话题 `/mission_command`。构造 `Multi_RflySim` 时读取 `./prompts/basic_cn.txt` 作为提示，加载本地模型 `./FM9G4B-V`（`trust_remote_code=True`，bfloat16，SDPA），放到 CUDA，先用提示预热上下文。`ask()` 流式生成并记录历史，`extract_code()` 抓首个代码块（可去掉 `command` 前缀），`process()` 返回代码块或原文。循环从 stdin 读指令并发布到 ROS；缺点：无异常/超时处理、`mission` 可能是 list 不是 string、无 spin/频率控制。
- `prompts/basic_cn.txt` / `prompts/system_promt.txt`：中文动作/回复格式提示（文件编码非 UTF-8），列出 takeoff、liftoff、pass_frame1 等动作，要求回答包含"思路/分步命令"并用 `command` 代码块包裹，供模型作为上下文。
- `WinWSL2.bat`：批处理，设置 `PSP_PATH`（默认 `C:\PX4PSP`），启动 VcXsrv（若未运行），切回脚本目录，进入 WSL 发行版 `WinWSL2-GPU`，用于提供 GUI/GPU 环境。
- 模型目录 `FM9G4B-V`（自定义 HuggingFace 多模态包）：
  - `config.json`：定义 LLM+视觉架构，62 层、hidden_size=2560、40 头、bfloat16、longrope，SigLIP 视觉（27 层，patch 14，image_size 980，slice_mode）。
  - `configuration_fm9g.py`：`FM9GConfig` / `FM9GVConfig` 配置类，含 RoPE、LoRA rank、缓存策略等。
  - `modeling_fm9g.py`：LLM 主体（基于 GPT-NeoX/OPT 改），支持 flash_attn/SDPA、GQA/MQA、KV cache、CausalLM 头等。

    - `modeling_navit_siglip.py`：SigLIP 视觉编码器（Navit 变体），支持 flash_attn。
    - `modeling_fm9gv.py`：多模态融合，语言模型 + 视觉模型，经 `Resampler` 将视觉特征映射为查询 token，并提供流式推理接口。
    - `resampler.py`：高分辨率视觉切片/重采样工具，生成窗口坐标，执行 RoIAlign 与拼接。
    - `image_processing_fm9gv.py` / `processing_fm9gv.py`：图像预处理与 Processor 封装，支持多尺寸图片、padding/truncation、batch_decode。
    - `tokenization_fm9g.py` / `tokenization_fm9gv_fast.py`：自定义 tokenizer，定义多模态特殊 token 与 chat 模板逻辑。
    - 其余 JSON/权重：`pytorch_model.bin` 模型权重，`tokenizer*.json` / `tokenizer.model` 词表等。
- 整体机制：本地加载多模态模型（文本+图像），结合中文提示，将用户输入映射为命令（优先代码块形式），再通过 ROS 发布执行。运行需 GPU、完整的 `FM9G4B-V`、提示文件，以及 ROS master 环境。

关键知识点3：多模态本地模型调用全链路

- 资源/文件：`FM9G4B-V/` 存放启元多模态权重与自定义 Transformers 代码（`modeling_fm9g*.py` 语言、`modeling_navit_siglip.py` 视觉、`processing_fm9gv.py` 处理理器等）；`prompts/basic_cn.txt` 约束动作词表与回答格式；`test_rflysim.py` 提供调用与 ROS 发布示例。
- 模型加载：`test_rflysim.py` 用 `AutoModel/AutoTokenizer.from_pretrained("./FM9G4B-V", trust_remote_code=True, torch_dtype=torch.bfloat16, attn_impleme` 载入多模态模型并放到 CUDA。
- 上下文注入：启动时读取 `prompts/basic_cn.txt`，通过 `self.ask()` 先喂给模型作为知识前缀，后续多轮对话共享同一 `self.msgs` 历史。
- 推理接口：调用自定义 `model.chat(image=None, msgs=self.msgs, tokenizer=...)` 流式生成；`stream=True` 逐块返回文本，结束后把助手回复写回 `self.msgs` 保持记忆。
- 指令抽取与下发：`extract_code()` 截取首个 `...` 代码块（去掉前缀 `command`），`process()` 返回该指令；主循环将结果发布到 ROS 话题 `/mission_command`（`std_msgs/String`），形成"本地推理 → 指令生成 → 仿真/实飞接口"的闭环。

6.参考资料

1. Ollama 官方文档
2. RflySim 安装指南
3. WSL2 网络模式说明（NAT / Mirrored）
4. PX4 SITL/HITL 官方文档
5. Win 防火墙入站规则操作：`wf.msc` → 高级设置 → 入站规则

7.常见问题

Q1：ollama 无法列出或运行模型?

A1：`ollama list` 空或 `ollama run` 无响应，检查：1) `ollama serve` 是否在跑；2) 模型是否已下载完整（必要时重新 pull）；3) 首次拉取是否被代理/防火墙拦截；4) `netstat -ano | find "11434"` 确认端口监听。

Q2：WSL/CLI 调用大模型返回 Connection refused 或超时?

A2：多半是 11434 未监听或被占用。`netstat -ano | find "11434"` 查看端口，必要时重启 `ollama serve`。如果是远端调用，检查 `OLLAMA_BASE_URL/OLLAMA_HOST` 是否写错。

Q3：UDP 无 Ack 或报 `[Errno 101] Network is unreachable`?

A3：目标 IP 不可达或防火墙拦截。Mirrored 模式下用真实目标机 IP；NAT 模式可用 `/etc/resolv.conf` 的 nameserver 或默认路由 IP 作为宿主 IP。确认目标端口 5005/UDP 已监听，并在 Windows 防火墙放行对应端口/程序。

Q4：WSL 没有 `/etc/resolv.conf` 或 `ip route show default` 为空?

A4：WSL 网络栈异常。执行 `wsl --shutdown`（管理员 PowerShell），重开 WSL；必要时删除残留的 `/etc/resolv.conf`，重启后应自动生成，`ip addr` 应能看到 eth0/eth1、`ip route show default` 应有网关。

Q5：防火墙整体关闭不起作用，但单独放行后恢复通信?

A5：流量匹配当前 Profile（Domain/Public/Private），默认规则仍拒绝；新建入站允许规则覆盖对应 Profile 后才放行。用 `netstat advfirewall show currentprofile` 确认当前生效的 Profile，针对 5005/UDP 或目标程序添加允许规则。

Q6：一键起飞等中文指令未触发?

A6：确认 LLM 返回的指令数组首项被映射到 `once_mode/TakeOff_mode`。当前代码已内置中文别名；若模型输出异常，可在 `.env`/环境变量里切换模型或降低温度，或在提示词中明确要求输出格式。

---

1. https://rflysim.com/ ↩
2. 推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf ↩