大模型多机避障测试

1. 实验目的

验证基于大模型自然语言/语音指令在室内多架旋翼无人机上进行避障规划

2. 实验要求

- 软件要求：Windows 10及以上版本；RflySim工具链[1]。
- 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\6.RflySimExtCtrl\3.CustExps\e1.LLM_CtrlUAVExps\5.ObsAvoidance

- `OpenAI_api_class.py`：主程序：大模型API调用
- `OpenAI_audio.py`：通用的实时语音识别模块
- `Config.xml`：航点配置文件
- `ReadConfig.py`：解析大模型API配置
- `Description.py`：提示词配置
- `Tool_Class.py`：轨迹可视化工具
- `udp_sender_test.py`：UDP通信工具
- `smoothAvoid_LLM_init.m`：MATLAB/Simulink外部控制程序
- `RflyUdpMavlinkRealSim.bat`：PX4 SITL 启动脚本

4. 实验内容或步骤

4.1 步骤1：大语言模型接口API配置

本实验已改为从 `\文档\Ogre\.env` 文件中读取 API 配置参数,配置方法与前序实验一致。完整配置说明可参考实验 `[RflySim 安装目录]\RflySimAPIs\6.RflySimExtCtrl\3.CustExps\e1.LLM_CtrlUAVExps\1.CommandCtrlUAVSim\Readme.pdf` 。

4.1.1 获取API密钥

需要在火山引擎平台获取两个模型的API密钥:

1. DeepSeek-V3 模型(浅层LLM)

进入火山引擎官网 https://www.volcengine.com ,在顶部工具栏的「大模型」菜单中点击「豆包大模型」,登录后进入「体验中心」→「语言模型」,切换为 **DeepSeek-V3**,点击「API 接入」创建并记录 **API Key**、**接口地址**和 **model 参数值**。

2. 豆包深度思考模型(深层LLM)

在火山引擎控制台,进入「大模型」→「深度思考模型」,选择 **doubao-1-5-thinking-pro-250415**,点击「API 接入」创建并记录对应的 **API Key**、**接口地址**和 **model 参数值**。

4.1.2 编辑配置文件

打开 `\文档\Ogre\.env` 文件(如不存在请手动创建),按以下格式填写双层大模型配置:

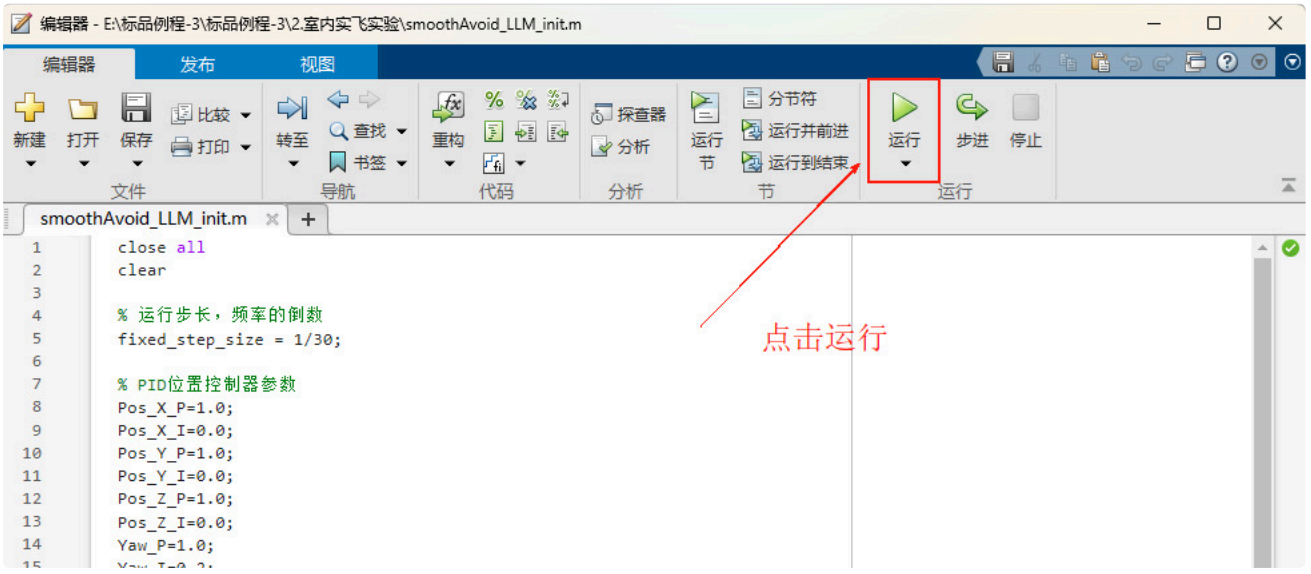
```
# 浅层LLM配置(DeepSeek-V3 - 快速响应)
VOLC_API_KEY=你的DeepSeek-V3_API_Key
VOLC_NAME=deepseek-v3-250324
VOLC_ENDPOINT=https://ark.cn-beijing.volces.com/api/v3

# 深层LLM配置(豆包深度思考 - 强化推理)
VOLC_DOUBAO_API_KEY=你的豆包深度思考_API_Key
VOLC_DOUBAO_NAME=doubao-1-5-thinking-pro-250415
VOLC_DOUBAO_ENDPOINT=https://ark.cn-beijing.volces.com/api/v3
```

保存并关闭配置文件。程序运行时会自动从该文件读取配置并初始化双层大模型客户端。

> **注意**: 原 `Config.xml` 配置文件已弃用,所有API配置统一通过 `.env` 文件管理。`Config.xml` 仅保留用于读取航点坐标等非敏感配置数据。

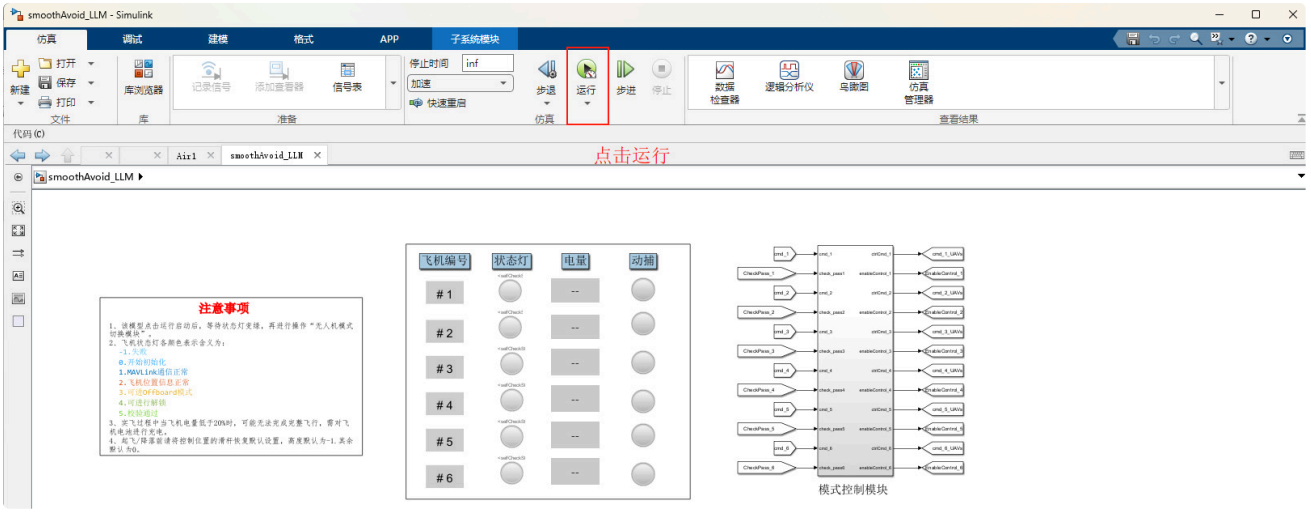
4.2 步骤2：软件在环仿真

在matlab中打开： smoothAvoid_LLM_init.m ；点击运行进行参数初始化



在matlab中打开smoothAvoid_LLM_sim.slx;

运行smoothAvoid_LLM_sim.slx



在VSCode中打开：`./ServerFile/OpenAI_api_class.py`;并运行

⑤首先，在终端输入指令："所有飞机起飞"，并回车；之后六架飞机将一起起飞

```
请输入你的控制指令：所有飞机起飞
======>> 进入对话模式 <<======
prompt: 所有飞机起飞
======================>> 所有飞机起飞模式： 开启Offboard模式
[Ack] b'U\xaa\x01\x00\x01\x00'
[Ack] Success
======================>> 所有飞机起飞模式： 开启解锁模式
[Ack] b'U\xaa\x01\x00\x01\x00'
[Ack] Success
======================>> 所有飞机起飞模式： 开启起飞模式
[Ack] b'U\xaa\x01\x00\x01\x00'
[Ack] Success
self.UDP_Config: {'IP': '127.0.0.1', 'Port': 5005, 'Send_Interv
 (0, 0, -1.2, 0), (0, 0, -1.2, 0), (0, 0, -1.2, 0)]}
执行耗时：11.853479 秒

正常Chat控制：ALL_FLY
```

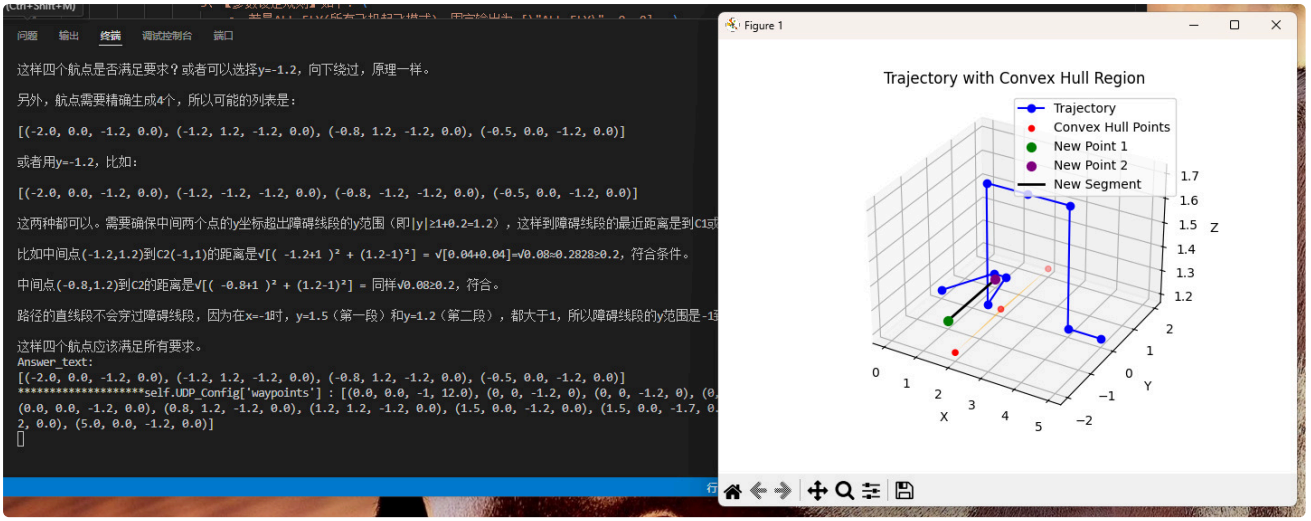
该例程中无人机有两种控制模式用于躲避空中的其他无人机：

- 第一种是任务执行模式(Task_Execution):
  控制无人机按照提前规划的航路进行飞行测试；
- 第二种是航路飞行模式(Point_Fly):
  控制无人机自主地飞往目标点并执行任务，无人机的航路将由大语言模型进行推理得到

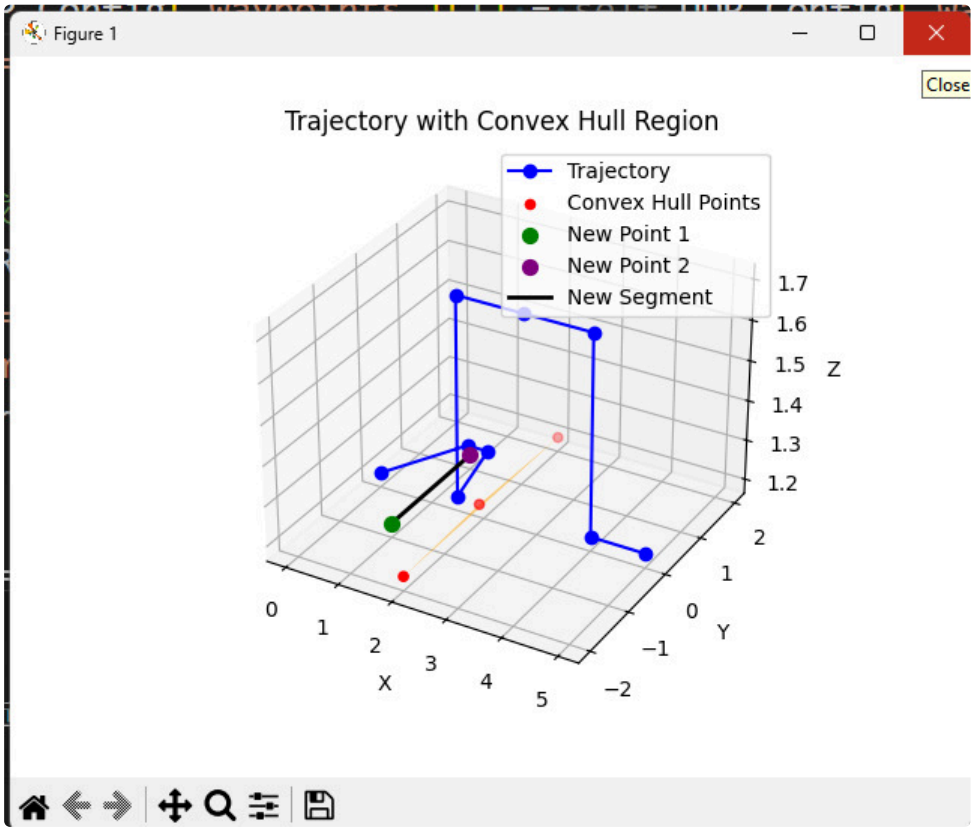
下面测试第一种模式，在终端种输入指令："任务执行模式"；无人机将会根据预先设定的路径进行飞行，最终到达终点

```
请输入你的控制指令：任务执行模式
======>> 进入对话模式 <<======
prompt: 任务执行模式
[Ack] b'U\xaa\x03\x00\x01\x00'
[Ack] Success
self.UDP_Config: {'IP': '127.0.0.1', 'Port': 5005,
(0.0, 0.0, -1.2, 0.0), (0.5, 1.8, -1.2, 0.0), (1.5
 -1.2, 0.0), (4.5, 0.0, -1.2, 0.0), (4.5, 0.0, -1.
执行耗时：1.639421 秒

正常Chat控制：Task_Execution
```

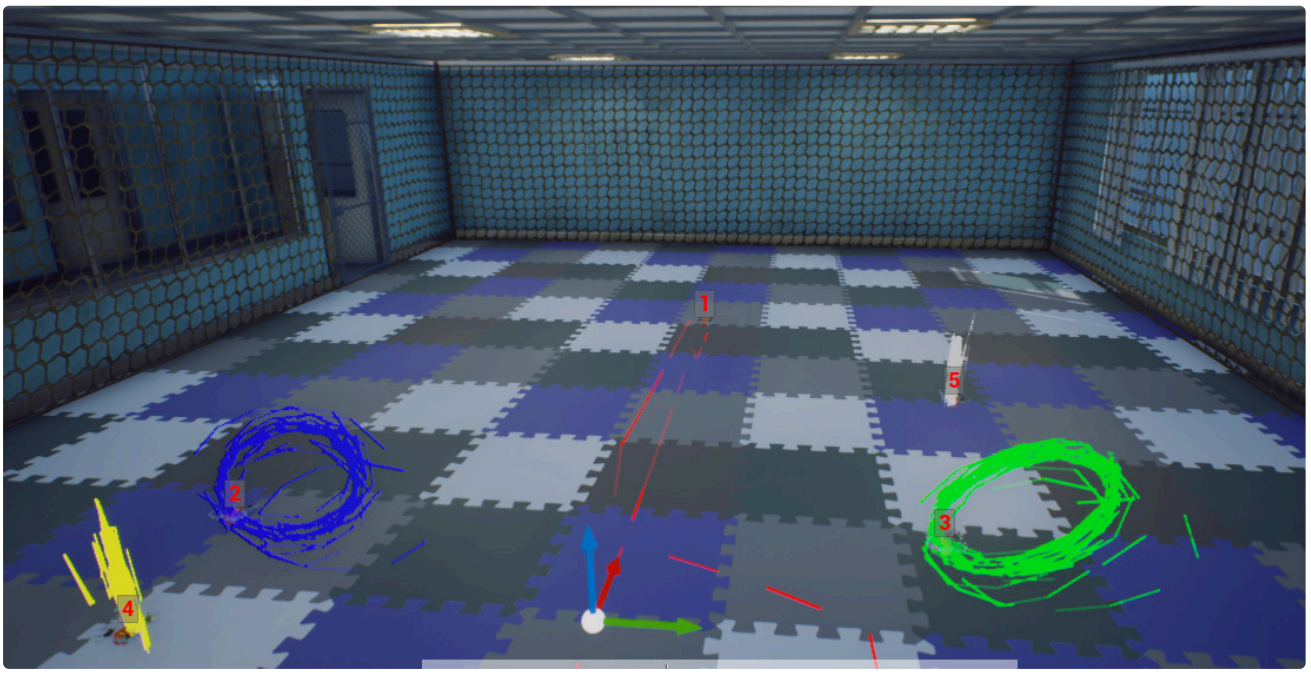
下面来测试第二种模式，在终端种输入指令："航路飞行模式"；之后终端中将会展示大语言模型的推理过程，最后输出计算推理得到的轨迹



关闭Python绘制的3D轨迹显示窗口，终端会询问是否采用该航路，输入"可以"，1号无人机将会根据前面推理出来的航线进行飞行

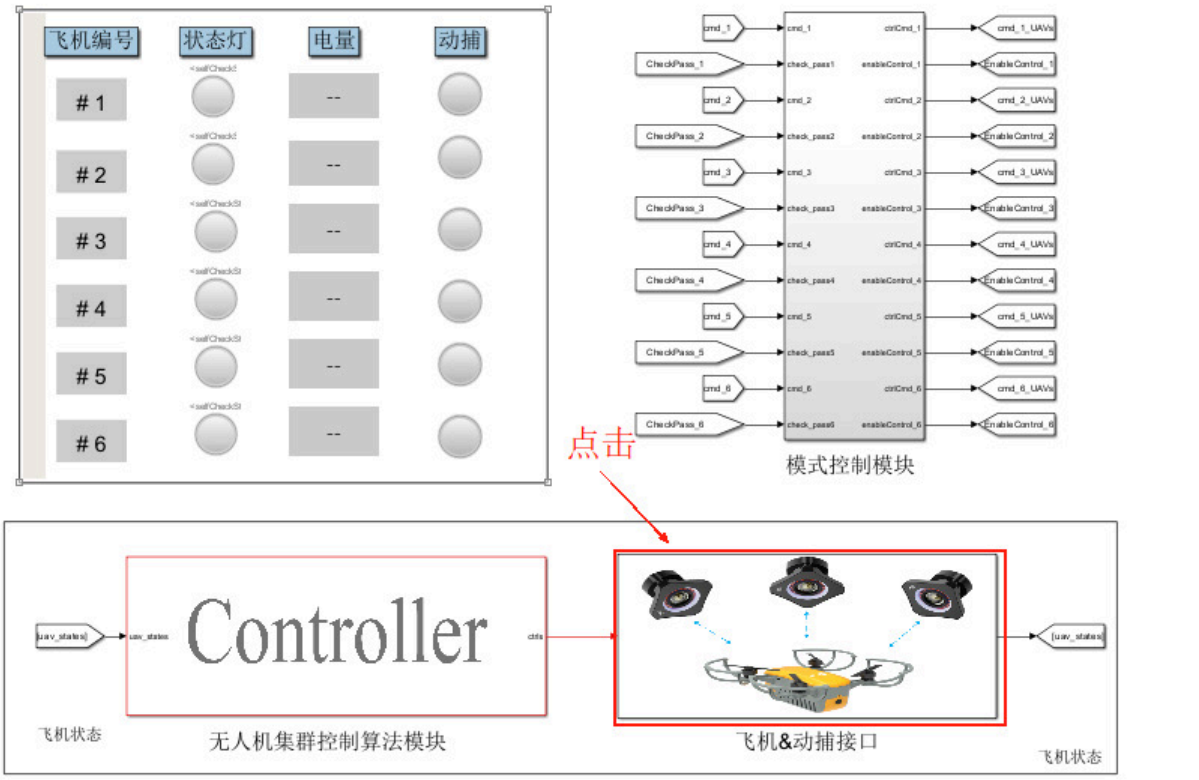


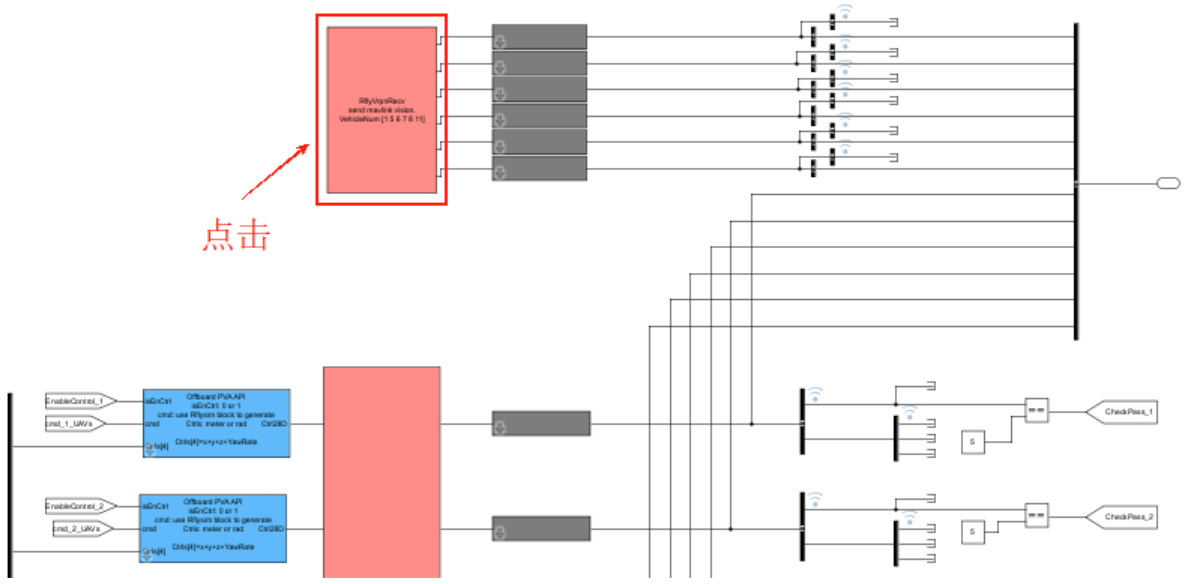
```
Answer_text:
[(-2.0, 0.0, -1.2, 0.0), (-1.2, 1.2, -1.2, 0.0), (-0.8, 1.2, -1.2, 0.0), (-0.
**********self.UDP_Config['waypoints'] : [(0.0, 0.0, -1, 12.0), (4.
-1.2, 0.0), (0, 0, -1.2, 0.0), (0, 0, -1.2, 0.0), (0, 0, -1.2, 0.0), (0.0, 0.
2, -1.2, 0.0), (1.5, 0.0, -1.2, 0.0), (1.5, 0.0, -1.7, 0.0), (2.75, 0.0, -1.7
, 0.0), (5.0, 0.0, -1.2, 0.0)]

是否采用此规划出的航路：可以

正常Tool调用：航路规划
[Ack] b'U\xaa\x03\x00\x01\x00'
[Ack] Success
======>> 航路点飞行
self.UDP_Config: {'IP': '127.0.0.1', 'Port': 5005, 'Send_Interval': 0.01, 'pa
), (5.0, 0.0, -1.2, 0.0), (0, 0, -1.2, 0.0), (0, 0, -1.2, 0.0), (0, 0, -1.2,
.0, 0.0, -1.2, 0.0), (0.8, 1.2, -1.2, 0.0), (1.2, 1.2, -1.2, 0.0), (1.5, 0.0,
, -1.7, 0.0), (4.0, 0.0, -1.7, 0.0), (4.0, 0.0, -1.2, 0.0), (5.0, 0.0, -1.2,
0.0), (5.0, 0.0, -1.2, 0.0), (5.0, 0.0, -1.2, 0.0)]}
执行耗时：116.929774 秒

正常Chat控制：Point_Fly
```

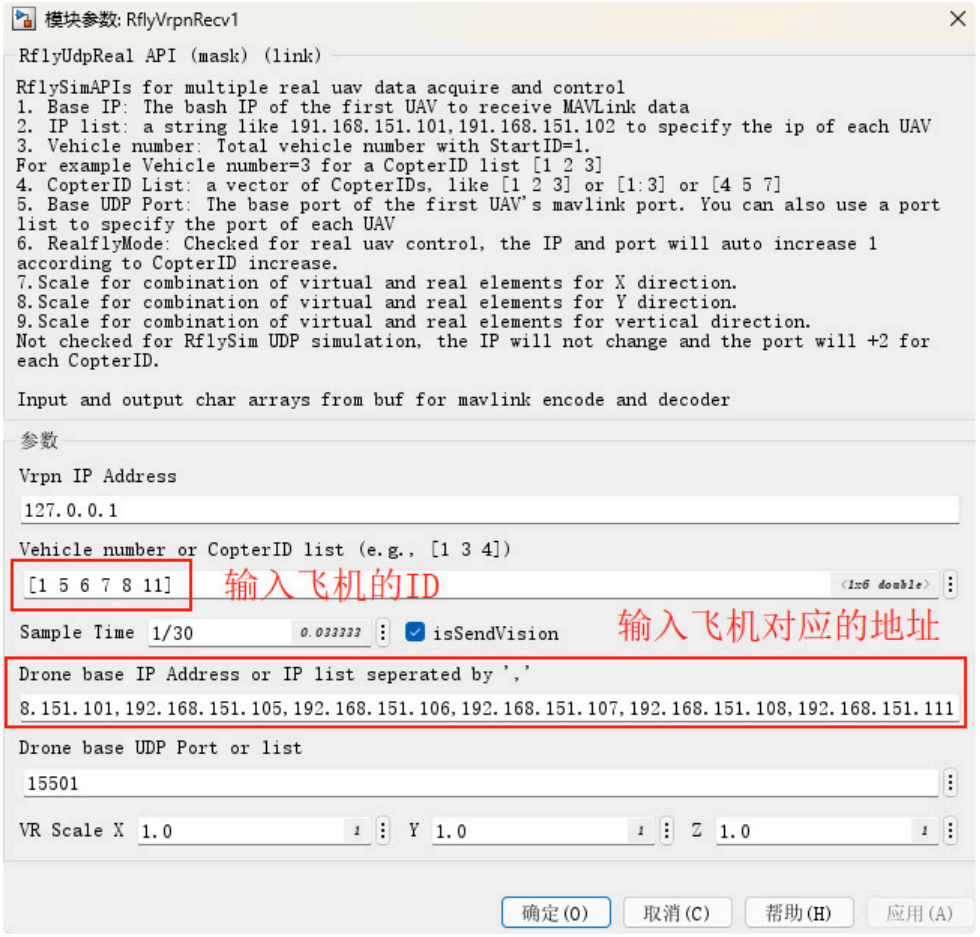


4.3 步骤3：室内实飞配置（选做）

按照 室内实飞实验.pdf 准备6架FSJ150飞机，在matlab中打开smoothAvoid_LLM.slx并配置每架飞机的通信地址，其余步骤与4.2 软件在环仿真相同：


飞机编号
状态灯
电量
动捕
1
2
3
4
5
6
模式控制模块
点击
Controller
飞机状态
无人机集群控制算法模块
飞机&动捕接口
飞机状态


点击



5. 关键知识点

5.1 关键知识点1：大模型驱动的避障路径规划

对应 `./ServerFile/Description.py` 中的 `Description.Make_Prompts_Calling_V2_Start` 与 `./ServerFile/OpenAI_api_class.py` 中的 `OpenAI_APIs.PointPromptRun`：不再依赖传统 A*/RRT 等几何算法，而是由 LLM 根据起点 A、终点 B 与障碍线段 C1–C2 的几何信息，直接生成满足安全约束的 4 个三维航点序列，由 `PointPromptRun` 解析后写入 `self.UDP_Config["waypoints"]` 供后续飞行执行。

5.2 关键知识点2：语义级任务描述与空间约束联合理解

对应 `./ServerFile/Description.py` 中的 `Description.Make_Prompts_Calling_V2_Start` 与 `./ServerFile/OpenAI_api_class.py` 中的 `OpenAI_APIs.GetPointChatMessage`：用户通过自然语言描述"从起点飞到气球上方"等任务，程序在 `GetPointChatMessage` 中自动计算起点、终点、与障碍墙的交点/中点 `PointIte_middle`，再在提示词中编码"绕过 C1–C2 线段、最近距离≥0.2 m、z=-1.2、yaw=0"等约束，让大模型同时理解任务语义与空间约束。

5.3 关键知识点3：路径转指令与多机协同机制

对应 `./ServerFile/OpenAI_api_class.py` 中的 `OpenAI_APIs.PointPromptRun`、`OpenAI_APIs.UDP_Send_Pro` 与 `./ServerFile/Tool_Class.py` 中的 `Tool_Class.show_img`：LLM 输出的 4 个规范化航点先被转换到以 1 号机为参考的局部坐标并追加到 `waypoints`，再在 `PointPromptRun` 中拼接中间过渡点/终点等关键航点，通过 `UDP_Send_Pro` 周期发送给 PX4，实现 1 架目标机在 5 架干扰机（构成障碍墙）之间的绕障飞行，并在 `Tool_Class.show_img` 中与障碍点集一起以 3D 轨迹形式可视化对比。

5.4 关键知识点4：双层大模型架构与职责分工

对应 `./ServerFile/OpenAI_api_class.py` 中的 `OpenAI_APIs.init_openai_clients`、`OpenAI_APIs.Chat` 与 `OpenAI_APIs.PointPromptRun`：`init_openai_clients` 同时初始化浅层 LLM（DeepSeek-V3，`self.client` + `self.LLMModel`）与深层 LLM（豆包深度思考模型，`self.client_deep` + `self.LLMModel_deep`）；浅层 LLM 在 `Chat` 中负责解析自然语言控制指令并映射成四类控制模式（起飞/降落/任务执行/航路飞行），深层 LLM 在 `PointPromptRun` 中专注于复杂障碍场景下的高精度航路规划，两者协同完成"语义理解 + 运动规划 + 实际控制"。

6.参考资料

1. RflySim官方文档
2. 大模型接口配置说明
3. PX4飞行器操作系统文档

7.常见问题

Q1：运行程序时出现API密钥错误或无法连接大模型?

A1：请检查 `\文档\Ogre\.env` 文件中的API配置是否正确，确认VOLC_API_KEY、VOLC_NAME和VOLC_ENDPOINT等参数值是否准确填入。另外，请确保网络连接正常且API服务可用。

Q2：无人机无法按预期路径飞行或避障效果不佳?

A2：这可能是由于大模型生成的航点不够精确或者无人机控制系统响应延迟导致的。可以尝试调整提示词中的约束参数，如增加与障碍物的最小距离要求，或检查Simulink模型中的控制参数是否合适。

Q3：如何评估避障算法的效果?

A3：可以通过观察无人机实际飞行轨迹与预期路径的偏差、避障过程中的安全性（与障碍物的距离）、以及任务完成时间等指标来评估算法效果。在仿真环境中，可以利用Tool_Class.py中的可视化功能来直观比较规划路径与实际飞行路径。

---

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