**“垂起固定翼无人机海上自主探索救援”参赛选手手册**

组织单位、参赛对象、比赛日程等相关信息，详见第十九届“挑战杯”全国大学生课外学术科技作品竞赛2024年度“揭榜挂帅”专项赛官方网站。

1. **答题要求**
2. 挑战赛（仿真赛）
3. 作品形式：

材料文档：内容包括但不限于方案作品设计报告、测试报告、作品录屏视频、总结报告和使用说明等文档。

软件模块：方案作品的源代码、可执行程序等，完成作品在指定仿真软件上的部署和运行。

作品考核：

综合考察信息融合、信息决策、无人机控制领域知识，要求参赛选手具备信息获取、信息处理、信息融合与决策的能力。选手需要通过自主编写智能算法程序，实现在一定区域内，利用智能算法操控虚拟无人机自主完成任务。

考核背景：

某海域，海警收到海面某遇险船只求救信号，该海域有多条船只正在进行海面作业，通过比对，海警大概了解船只的外形特征，但是对于遇险船只具体位置、遇险情况均未知。

遇险船只海域伴有雷暴等极恶劣天气状况，海警救援无人机无法进行遥控操作，只能采取智能自主飞行。海警按照遇险船只发出的救援信号GPS定位范围信息，迅速派出救援无人机前往遇险船只海域范围内，搜寻遇险船只，降落到遇险船只上进行救援。

挑战赛（仿真赛）构建了该海域的仿真环境（仿真环境详见第四项环境配置及安装程序），选手利用智能算法操控虚拟无人机自主完成任务。虚拟无人机从起飞平台起飞至GPS引导点，到达GPS引导点后，通过待救援目标发出的带有噪声的航向信息寻找真目标，然后再通过视觉引导锁定真目标后降落完成整个比赛过程。

1. 参赛选手在挑战赛阶段需在指定的仿真软件上展开算法调试并完成方案设计；
2. 参赛选手需要提供具体的算法描述，并自行组织对软件设计进行合理性评估；
3. 参赛选手必须确保其作品的独立原创性，严禁抄袭或剽窃他人成果，并且必须遵守国家相关的知识产权保护规定。参赛选手不得侵犯任何第三方的知识产权或其他权益。如若发生知识产权纠纷，参赛者承担相应责任；
4. 参赛选手提交的材料原则上不予退还，请参赛选手自行保存底稿；

作品已获得国际竞赛、国家级奖励和其他全国性竞赛获奖作品的，不在申报作品范围之列。

1. 擂台赛（实物赛）

1. 作品形式：

资料文档：内容包括但不限于方案作品设计报告、测试报告、总结报告和使用说明等文档；

软件模块：方案作品的源代码、可执行程序等；

作品要求：

参赛选手在擂台赛（实物赛）阶段需在指定场地、指定无人机上展开算法调试并完成实际飞行任务；

1. **任务概要**

（一）挑战赛（仿真赛）

1. 赛题：

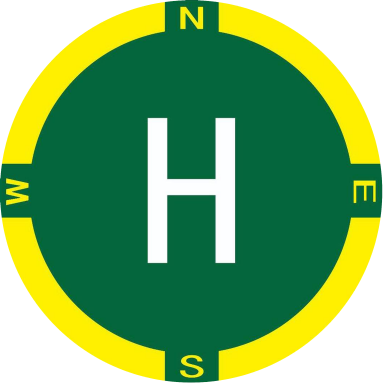
挑战赛（仿真赛）构建了该海域的仿真环境，选手利用智能算法操控虚拟无人机自主完成任务。虚拟无人机从起飞平台起飞至GPS引导点，到达GPS引导点后，通过待救援目标发出的带有噪声的航向信息寻找真目标，然后再通过视觉引导锁定真目标后降落完成整个比赛过程。

开始解题后，建议虚拟无人机起飞至25米高度后，将虚拟无人机切换为固定翼模式飞行。飞行过程中，虚拟无人机需要避开雷暴区域（固定区域内随机形成），根据GPS提供的位置信息飞行至GPS引导点后，开始寻找真目标。

在GPS引导点直径一公里范围内，有8个假目标和1个真目标，通过接收真目标（待救援船只）发布的救援信号，靠近真目标（待救援船只）位置。由于海浪、风向等原因，真目标（待救援船只）的位置会不断发生变化，虚拟无人机需要通过视觉识别分辨真假目标，引导虚拟无人机锁定真目标（待救援船只）并降落。



真目标（待救援船只）示例图

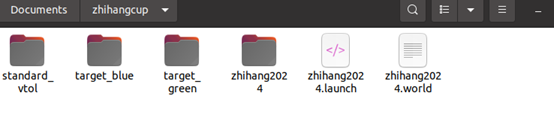


真目标（待救援船只）降落点示例图

1. 解题

选手可通过发送邮件（邮箱： 18210263093@163.com），获得仿真环境安装教学视频和比赛赛题相关文件。

比赛赛题相关文件内容如下：



请选手按照比赛官方教程配置好环境后将赛题文件夹中的文件放入相应位置:

1. 模型文件：standard\_vtol,target\_blue,target\_green放置位置为：

~/PX4\_Firmware/Tools/sitl\_gazebo/models 若放置时提示文件夹内已经有同名文件，直接将同名文件替换即可。

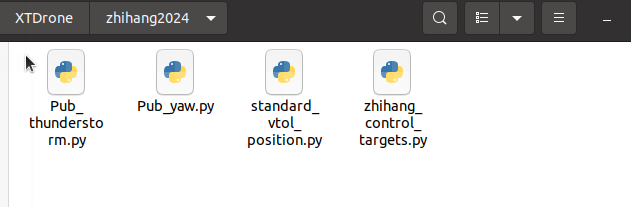
1. Launch文件：zhihang2024.launch放置位置为:

~/PX4\_Firmware/launch若放置时提示文件夹内已经有同名文件，直接将同名文件替换即可。

1. World文件：zhihang2024.world放置位置为:

~/PX4\_Firmware/Tools/sitl\_gazebo/worlds 若放置时提示文件夹内已经有同名文件，直接将同名文件替换即可。

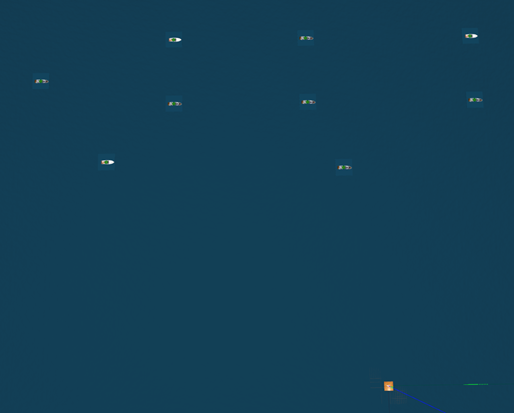
1. 其他文件夹: zhihang2024放置位置为：~/XTDrone 若放置时提示已有同名文件夹直接替换即可。zhihang2024文件夹中的文件如下:



* 注意：比赛官方提供的所有文件都不得随意更改，若更改则判定选手的成绩无效。

1. 操作步骤:
2. 启动仿真程序:

roslaunch px4 zhihang2024.launch



1. 运行通信脚本：

cd ~/XTDrone/communication/

python vtol\_communication.py standard\_vtol 0

1. 启动发布位置代码：

cd ~/XTDrone/zhihang2024

python standard\_vtol\_position.py

启动需要救援的船只发布角度代码：

cd ~/XTDrone/zhihang2024

python Pub\_yaw.py

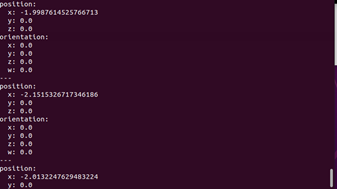
启动发布雷暴区中心点的代码：

cd ~/XTDrone/zhihang2024

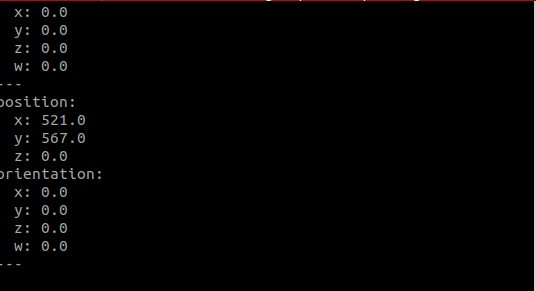
python Pub\_thunderstorm.py

选手不得改动上述代码，仅能通过订阅/zhihang/standard\_vtol/angel话题来获得角度信息。其中position下的x为角度信息，采用弧度制。

同时，选手可以通过订阅/zhihang/thunderstorm话题来获得雷暴中心点的信息，其中position下的x为雷暴中心点x，position下的y为雷暴中心点y。（单位：米）



订阅/zhihang/standard\_vtol/angel内容图



订阅/zhihang/thunderstorm内容图

1. 记录数据，要求选手在无人机起飞前启动数据记录：

cd ~/XTDrone/zhihang2024

rosbag record -O score1 /standard\_vtol\_0/mavros/state /gazebo/model\_states /xtdrone/standard\_vtol\_0/cmd /ship /arrive /zhihang/thunderstorm

* 注意：选手需严格按照比赛方要求步骤启动程序，比赛过程中不能随意关闭或重新开启以上程序，否则可能会导致得分无效。

选手需要保存生成的score1.bag文件，压缩后按主办放提供的方式上交作品。

（二）擂台赛（实物赛）

1. 赛题

无人机从起飞平台起飞至GPS引导点范围内，在GPS引导点范围内通过视觉引导及待救援目标发出的救援信号寻找真目标，锁定真目标后空投物体到真目标降落点，空投完成后返航完成整个比赛过程。

★特等奖获奖队伍必须参加擂主赛（实物赛），否则视为放弃奖项，获奖名次依次递补。无人机调试时间为一周时间，由主办方提供无人机及试飞区域。实物赛使用无人机及比赛参数详见《“垂起固定翼无人机海上自主探索救援”实物赛参赛选手手册》。

1. **作品评判标准**
2. 挑战赛（仿真赛）

评委主要按照作品的符合性、作品的完整性、系统得分进行综合评价，分值分配情况如下:

1.作品符合性：10分

研究思路合理性(分值:5分)

技术路线可行性(分值:5分)

2.作品完整性：10分

作品的源代码、可执行程序等可以在赛方提供的仿真环境进行部署和正确运行(分值:10分)

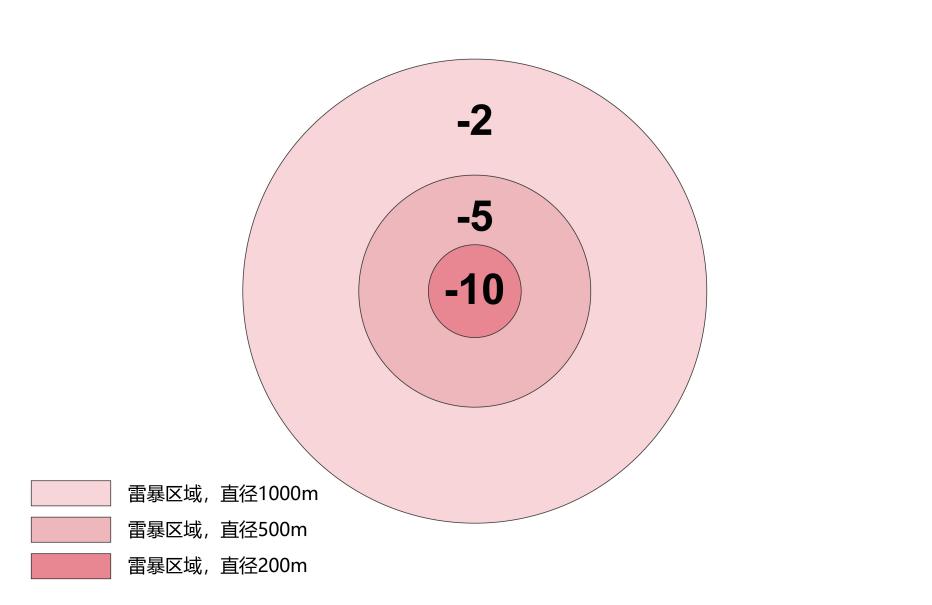
3.系统得分：80分

阶段一(20分)：虚拟无人机从起飞平台起飞至GPS引导点范围内（GPS引导点范围直径6米），发布确认到达信息。本阶段按照选手躲避雷暴区域的路径规划及完成任务用时计算分数。

虚拟无人机起飞后，需要切换为固定翼模式飞行，否则本阶段得分为0。

雷暴区域直径1000米，由3个同心圆组成。

虚拟无人机飞行轨迹（虚拟无人机在地图上X轴与Y轴的交汇点轨迹）经过同心圆中心部分直径200米，扣10分，经过外直径200米-500米，扣5分，经过直径500米-1000米，扣2分。没有经过雷暴区域不扣分。



雷暴区三个同心圆示例图

GPS引导点范围直径6米，未到达直径范围内，发布到达信息，本阶段得分为0。

本阶段按照选手完成任务用时T1(单位s)计算分数，计算公式为：T1 = 发送到达GPS引导点范围内（引导点范围直径6米） – 虚拟无人机旋桨解锁并起飞离开起飞平台的时间。

当T1<=420s得10分;

当420s<T1<=480s得5分;

当480s<T1<=600s得2分，

T1>600s得0分。

阶段二(20分)：在GPS引导点范围内，有8个假目标和1个真目标。虚拟无人机通过接收真目标（待救援船只）发布的救援信号，靠近真目标（待救援船只）位置。由于海浪、风向等原因，真目标（待救援船只）的位置会不断发生变化，虚拟无人机需要通过视觉算法分辨真假目标，引导虚拟无人机锁定真目标（待救援船只）并降落。本阶段按照选手到达真目标点范围及分辨真假目标用时计算分数，未到达真目标点范围内或超过阶段规定时间范围，则得分为0分。

虚拟无人机根据真目标（待救援船只）发布的救援信号靠近真目标，通过视觉区分真假目标最后锁定真目标，锁定真目标后，发布确认到达信息。（到达信息方式为：发布/arrive话题，该话题的消息类型为Bool，发布消息内容为True即告知对接开始。）

到达真目标（待救援船只）目标点直径10米范围内并识别出真目标时，发布确认真目标信息。（真目标确认方式：发布/ship话题，该话题的消息类型为Bool，发布消息内容为True即告知对接开始。）

完成时间T2(单位s)来计算分数，计算公式为：T2 = 发送真目标确认信息 – 第一阶段发送确认信息时间，

当T2<=120s得20分；

当120s <T2<=240s得15分；

当240s <T2<=360s得10分，

T2>360s得0分。

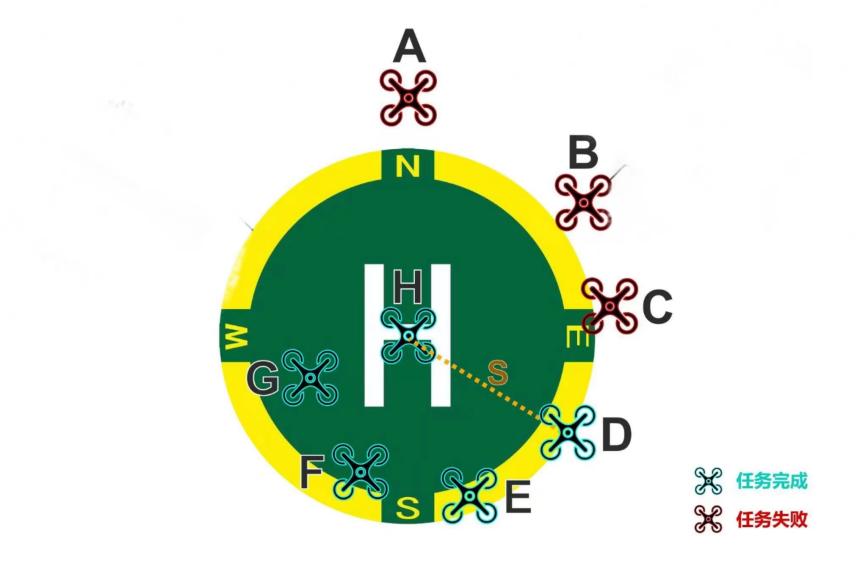
阶段三(40分)：虚拟无人机的降落方式默认为旋翼模式降落，锁定虚拟无人机桨叶转动即视为降落。本阶段按照选手完成任务用时及虚拟无人机锁桨并与降落点的水平距离计算分数。选手需在规定时间内完成任务取得分数。

虚拟无人机与海面高度不高于5米（注意：目标船只高度3.5米），否则本阶段得分为0分。

虚拟无人机从起飞平台起飞至锁桨降落总时长T3。当T3<=1200s，不扣分；当T3>1200s，系统得分为0分。

虚拟无人机以固定翼模式连续飞行时间GT>=20s，否则系统得分得分为0分。

虚拟无人机锁桨降落时与降落区中心点距离远近评分，不得降落在降落区外，否则本阶段得分为0分。



（示例图）

通过虚拟无人机与目标中心点的距离D(单位：米)来计算分数，计算公式为:本阶段得分G=(100-33xD)x0.5。

当D> 3米 时，本阶段得分为0分。

* 选手得分相同时，根据用时T3进行排名。（T3时长取小数点后四位）

1. 擂台赛（实物赛）

评委主要按照现场答辩、实物飞行2方面进行综合评价，分值分配情况如下:

1.现场答辩表现：（10分）

答辩时需要向评委说明作品设计方案、作品实现方式等内容，根据答辩实际情况综合评定。

2.实物飞行表现：（90分）

1. 阶段一：无人机根据GPS引导飞行到指定区域（分值:20分）
2. 阶段二：无人机按照信号和图像识别，飞行至指定真目标上方（分值:20分）
3. 阶段三：无人机投掷物品到降落区的精准度（分值:40分）
4. 阶段四：无人机返航（分值:10分）

★特等奖获奖队伍必须参加擂主赛（实物赛），否则视为放弃奖项，获奖名次依次递补。无人机调试时间为一周时间，由主办方提供无人机及试飞区域。实物赛使用无人机及比赛参数详见《“垂起固定翼无人机海上自主探索救援”实物赛参赛选手手册》。

1. 环境配置及安装教程

仿真平台及参数

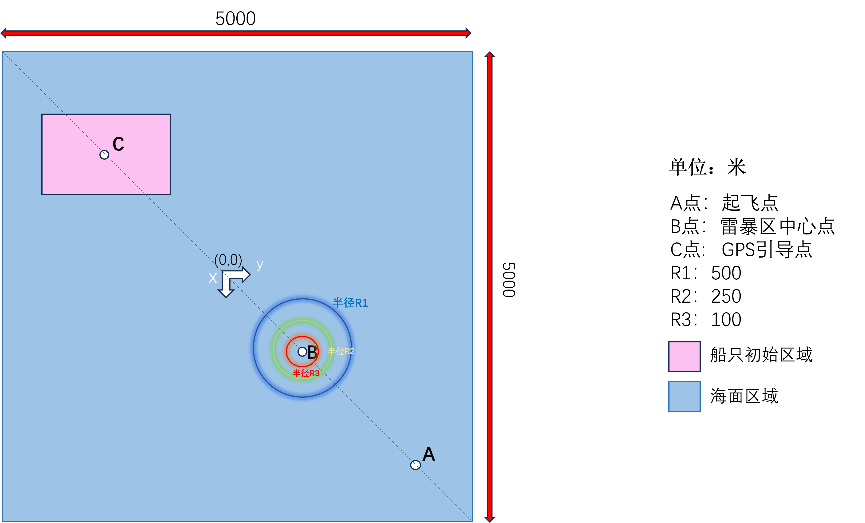
1. 操作系统

Ubuntu20.04

1. 仿真平台

Gazebo11，搭配QGC使用

1. 仿真地图



仿真环境海域示例图

面积：5千米×5千米=25平方千米

雷暴区域：雷暴区域由三个同心圆组成，其中大圆直径1000米，中圆直径500米，小圆直径200米。同心圆的圆心（B点）在一个正方形区域内随机生成，正方形中心点为(600,600)，边长为200米。

起飞点（A点）：起飞平台中心点位于(2300,2300)。

GPS引导点（C点）：GPS引导点位于(-1200,-1200)，该点位于待救援船只中心区域附近。

虚拟无人机参数：

Standard\_vtol无人机在仿真环境中：长：0.55米，翼展：2.144米，高：0.05米，重量：5千克。

四旋翼模式推荐速度：3m/s

固定翼模式推荐速度：10-17m/s，转弯半径：FW\_L1\_PERIOD（L1控制器时间常数）推荐范围10-15s。

地面站参数详见：

http://docs.px4.io/v1.11/zh/advanced\_config/parameter\_reference.html

* 附录：px4开发环境搭建教程

px4开发环境搭建教程

注意：本教程系统使用的是ubuntu20.04

1.Ubuntu系统换源

由于Ubuntu20.04系统安装默认源为国外源（服务器位于国外），导致国内用户在开发的过程中会遇到一系列问题，为避免该系列问题，需要我们对安装好的Ubuntu系统进行换源操作。

打开终端依次输入：

备份---sudo cp /etc/apt/sources.list /etc/apt/[sources.list.back](http://sources.list.back/)

修改源---sudo vim /etc/apt/sources.list （选择添加的镜像源和这部分操作可看博客）https://blog.csdn.net/tangling1/article/details/132150597或者https://blog.csdn.net/qq\_45878098/article/details/126037838）

更新---sudo apt-get update

2.一键安装ros （此部分推荐使用该指令一键安装，不要看ros官网的安装步骤）

（1）所需要的指令只有一个，在终端输入：

wget http://fishros.com/install -O fishros && . fishros

安装过程中需要在[ ]输入数字，依次是1、2、1、1 就能安装成功（注意看是noetic版本）

（2）验证，开三个不同的终端分别输入指令：

roscore

rosrun turtlesim turtlesim\_node

rosrun turtlesim turtle\_teleop\_key

键盘运行小乌龟检测是否安装成功

3.依赖安装（只有三条指令，注意第二条指令是pip3不是pip2，三条指令复制完整）

sudo apt install ninja-build exiftool ninja-build protobuf-compiler libeigen3-dev genromfs xmlstarlet libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev python3-pip gawk

pip3 install pandas jinja2 pyserial cerberus pyulog==0.7.0 numpy toml pyquaternion empy pyyaml

pip3 install packaging numpy empy toml pyyaml jinja2 pyargparse kconfiglib jsonschema future

这部分如果出现报错，参考语雀教程，可先更新 setuptools 和 pip

4.Gazebo安装

（1）先新建工作站，在终端依次输入以下指令

mkdir -p ~/catkin\_ws/src

mkdir -p ~/catkin\_ws/scripts

cd catkin\_ws && catkin init

catkin build

（2）卸载之前的Gazebo，可在上一个终端往下操作

cd..

sudo apt-get remove gazebo\*

sudo apt-get remove libgazebo\*

sudo apt-get remove ros-noetic-gazebo\*

（3）安装gazebo

sudo sh -c 'echo "deb http://packages.osrfoundation.org/gazebo/ubuntu-stable `lsb\_relea se -cs` main"> /etc/apt/sources.list.d/gazebo-stable.list' （粘贴时注意这是一条完整的指令，中间没有换行）

cat /etc/apt/sources.list.d/gazebo-stable.list

wget https://packages.osrfoundation.org/gazebo.key -O - | sudo apt-key add -

sudo apt-get update（确定这条指令执行完成后再操作下面）

sudo apt-get install gazebo11

sudo apt-get install libgazebo11-dev

最后在终端输入指令：gazebo，若出现界面则成功安装

（4）Gazebo的ros插件

先安装依赖 （下面是一条完整的指令，没有回车）：

sudo apt-get install ros-noetic-moveit-msgs ros-noetic-object-recognition-msgs ros-noetic-octomap-msgs ros-noetic-camera-info-manager ros-noetic-control-toolbox ros-noetic-polled-camera ros-noetic-controller-manager ros-noetic-transmission-interface ros-noetic-joint-limits-interface

安装完成后依次输入：

cd ~/catkin\_ws

catkin build （这里会报错，出现failed表示不成功，不成功后可以输入指令cakin clean 再继续catkin build，多重复几次；）

编译成功后执行如下两条指令，判断gazebo\_ros是否安装成功：

打开第一个终端，输入roscore

打开第二个终端，依次输入：

source ~/catkin\_ws/devel/setup.bash

rosrun gazebo\_ros gazebo

出现gazebo界面表示成功；

最后还需要下载语雀文档里的models.zip，将该附件解压缩后放在~/.gazebo中（注意.gazebo是隐藏文件夹，需要ctrl+h显示出来）解压完成后在~/.gazebo/models/路径下可以看到很多模型

5.MAVROS安装

在新的终端依次输入以下四条指令

sudo apt install ros-noetic-mavros ros-noetic-mavros-extras

wget [https://gitee.com/robin\_shaun/XTDrone/raw/master/sitl\_config](https://gitee.com/robin_shaun/XTDrone/raw/master/sitl_config/mavros/install_geographiclib_datasets.sh)

[/mavros/install\_geographiclib\_datasets.sh](https://gitee.com/robin_shaun/XTDrone/raw/master/sitl_config/mavros/install_geographiclib_datasets.sh)

sudo chmod a+x ./install\_geographiclib\_datasets.sh

sudo ./install\_geographiclib\_datasets.sh #这步需要装一段时间

6.PX4配置

这里采用的方法是直接下载语雀文档里的压缩包再编译的办法。

语雀文档链接：

https://www.yuque.com/xtdrone/manual\_cn/basic\_config\_13

（1）先下载PX4\_Firmware.zip，解压后在终端依次输入：

cd PX4\_Firmware (确定进入到该文件夹下)

make px4\_sitl\_default gazebo（会报错ninja，解决办法先执行：make clean再执行该指令，编译完成后出现gazebo界面关闭即可）

（2）修改路径（在home中点开隐藏的~/.bashrc文件，在最下面添加如下四条路径，注意不要更改下面四个路径的顺序）：

source ~/catkin\_ws/devel/setup.bash

source ~/PX4\_Firmware/Tools/setup\_gazebo.bash ~/PX4\_Firmware/ ~/PX4\_Firmware/build/px4\_sitl\_default

export ROS\_PACKAGE\_PATH=$ROS\_PACKAGE\_PATH:~/PX4\_Firmware

export ROS\_PACKAGE\_PATH=$ROS\_PACKAGE\_PATH:~/PX4\_Firmware/Tools/sitl\_gazebo

添加完毕并保存后运行：source ~/.bashrc

再开个终端依次运行：

cd ~/PX4\_Firmware

roslaunch px4 mavros\_posix\_sitl.launch

出现gazebo界面里面有个无人机表示成功

（3）通信验证

再开一个终端运行命令：rostopic echo /mavros/state

若出现connected: True,则说明MAVROS与SITL通信成功。

如果是false，先确保.bashrc里的路径正确，如果确定正确还是false，在launch文件夹下找到 mavros\_posix\_sitl.launch文件点开，修改第25行的：nane="fcu ur1" default= ”udp: I/:145400locaLhost:14557"，把udp修改成“udp:“:245400LocaLhost: 34580”保存，并重新运行roslaunch px4 mavros\_posix\_sitl.launch和rostopic echo /mavros/state

7.QGC地面站安装

（1）新开一个终端依次输入以下5条指令：

sudo usermod -a -G dialout $USER

sudo apt-get remove modemmanager -y

sudo apt install gstreamer1.0-plugins-bad gstreamer1.0-libav gstreamer1.0-gl -y

sudo apt install libqt5gui5 -y

sudo apt install libfuse2 -y

（2）点开QGC官网：<https://docs.qgroundcontrol.com/master/en/qgc-user-guide/getting_started/download_and_install.html>

找到“Ubuntu Linux”处点击下载



下载完成后先在终端里cd到下载的文件夹里，依次输入下面两个指令：

chmod +x ./QGroundControl.AppImage

./QGroundControl.AppImage

安装完成后会跳出QGC的界面表示成功

8.XTDrone

（1）新开一个终端或者在上一个终端输入cd..，然后依次输入以下指令（下面的指令只要不报红色的错误都可以不管，继续往下进行）：

git clone https://gitee.com/robin\_shaun/XTDrone.git

cd XTDrone

git checkout 1\_13\_2

git submodule update --init --recursive

# 修改启动脚本文件

cp sitl\_config/init.d-posix/\* ~/PX4\_Firmware/ROMFS/px4fmu\_common/init.d-posix/

# 添加launch文件

cp -r sitl\_config/launch/\* ~/PX4\_Firmware/launch/

# 添加世界文件

cp sitl\_config/worlds/\* ~/PX4\_Firmware/Tools/sitl\_gazebo/worlds/

# 修改部分插件（这部分可能会显示缺东西什么的，可以不用管，尤其是后两条指令，报错可以继续往下进行）

cp sitl\_config/gazebo\_plugin/gimbal\_controller/gazebo\_gimbal\_controller\_plugin.cpp ~/PX4\_Firmware/Tools/sitl\_gazebo/src

cp sitl\_config/gazebo\_plugin/gimbal\_controller/gazebo\_gimbal\_controller\_plugin.hh ~/PX4\_Firmware/Tools/sitl\_gazebo/include

cp sitl\_config/gazebo\_plugin/wind\_plugin/gazebo\_ros\_wind\_plugin\_xtdrone.cpp ~/PX4\_Firmware/Tools/sitl\_gazebo/src

cp sitl\_config/gazebo\_plugin/wind\_plugin/gazebo\_ros\_wind\_plugin\_xtdrone.h ~/PX4\_Firmware/Tools/sitl\_gazebo/include

# 修改CMakeLists.txt

cp sitl\_config/CMakeLists.txt ~/PX4\_Firmware/Tools/sitl\_gazebo

# 修改部分模型文件

cp -r sitl\_config/models/\* ~/PX4\_Firmware/Tools/sitl\_gazebo/models/

# 替换同名文件

cd ~/.gazebo/models/

rm -r stereo\_camera/ 3d\_lidar/ 3d\_gpu\_lidar/ hokuyo\_lidar/

（2）重新编译，依次输入下面三条指令

cd ~/PX4\_Firmware

rm -r build/

make px4\_sitl\_default gazebo（编译时报错可以make clean再继续执行编译，出现报错可以多编译几次）

新开一个终端，输入指令：source ~/ . bashrd

9.验证环境是否可用

使用键盘控制无人机

在一个终端中运行以下指令：

cd ~/PX4\_Firmware

roslaunch px4 indoor1.launch

再开一个终端打开通信程序：

cd ~/XTDrone/communication/

python multirotor\_communication.py iris 0

再开一个终端运行：

cd ~/XTDrone/control/keyboard

python multirotor\_keyboard\_control.py iris 1 vel

如果室内场景能够成功加载、四旋翼可以通过键盘控制起飞，证明环境已经成功配置完成。