**无人机仿真环境配置方法**

**0. 仿真环境说明**

仿真环境使用的操作系统为Ubuntu 18.04，相关软件版本如下：

·ROS：ROS Melodic  
·Gazebo：Gazebo 9.0.0  
·PX4：v1.9.2

建议分配给操作系统的内存不小于4GB，否则Gazebo可能无法正常运行。

以下环境配置过程假设你已经完成了Ubuntu系统的安装。

**1. 安装ROS Melodic和Gazebo 9**

该部分安装过程与ROS官网安装教程基本一致，详情可参照http://wiki.ros.org/cn/melodic/Installation/Ubuntu，也可以参考课程之前的相关课件。此处不再赘述ROS的安装过程。Gazebo 9会随着ROS Melodic桌面完全版一同安装。如果你安装的ROS不是桌面完全版，可以执行如下命令安装Gazebo 9：

sudo apt install gazebo9 ros-melodic-gazebo\*

也可以前往Gazebo官网手动安装。

如果你使用的是虚拟机，需要额外执行以下命令以确保Gazebo能正常运行：

echo "export SVGA\_VGPU10=0" >> ~/.bashrc

source ~/.bashrc

**2. 安装PX4**

该部分安装过程可参照https://zhuanlan.zhihu.com/p/91329291和PX4官方网站。

首先进行toolchain的安装。创建文件ubuntu\_sim\_common\_deps.sh，向其中写入：

#!/bin/bash

## Bash script for setting up a PX4 development environment on Ubuntu LTS (16.04).

## It can be used for installing simulators (only) or for installing the preconditions for Snapdragon Flight or Raspberry Pi.

##

## Installs:

## - Common dependencies and tools for all targets (including: Ninja build system, Qt Creator, pyulog)

## - FastRTPS and FastCDR

## - jMAVSim simulator dependencies

## - PX4/Firmware source (to ~/src/Firmware/)

# Preventing sudo timeout https://serverfault.com/a/833888

trap "exit" INT TERM; trap "kill 0" EXIT; sudo -v || exit $?; sleep 1; while true; do sleep 60; sudo -nv; done 2>/dev/null &

# Ubuntu Config

echo "We must first remove modemmanager"

sudo apt-get remove modemmanager -y

# Common dependencies

echo "Installing common dependencies"

sudo apt-get update -y

sudo apt-get install git zip qtcreator cmake build-essential genromfs ninja-build exiftool astyle -y

# make sure xxd is installed, dedicated xxd package since Ubuntu 18.04 but was squashed into vim-common before

which xxd || sudo apt install xxd -y || sudo apt-get install vim-common --no-install-recommends -y

# Required python packages

sudo apt-get install python-argparse python-empy python-toml python-numpy python-dev python-pip -y

sudo -H pip install --upgrade pip -i https://pypi.tuna.tsinghua.edu.cn/simple/

sudo -H pip install pandas jinja2 pyserial pyyaml -i https://pypi.tuna.tsinghua.edu.cn/simple/

# optional python tools

sudo -H pip install pyulog -i https://pypi.tuna.tsinghua.edu.cn/simple/

# Install FastRTPS 1.7.1 and FastCDR-1.0.8

fastrtps\_dir=$HOME/eProsima\_FastRTPS-1.7.1-Linux

echo "Installing FastRTPS to: $fastrtps\_dir"

if [ -d "$fastrtps\_dir" ]

then

echo " FastRTPS already installed."

else

pushd .

cd ~

wget https://www.eprosima.com/index.php/component/ars/repository/eprosima-fast-rtps/eprosima-fast-rtps-1-7-1/eprosima\_fastrtps-1-7-1-linux-tar-gz -O eprosima\_fastrtps-1-7-1-linux.tar.gz

tar -xzf eprosima\_fastrtps-1-7-1-linux.tar.gz eProsima\_FastRTPS-1.7.1-Linux/

tar -xzf eprosima\_fastrtps-1-7-1-linux.tar.gz requiredcomponents

tar -xzf requiredcomponents/eProsima\_FastCDR-1.0.8-Linux.tar.gz

cpucores=$(( $(lscpu | grep Core.\*per.\*socket | awk -F: '{print $2}') \* $(lscpu | grep Socket\(s\) | awk -F: '{print $2}') ))

(cd eProsima\_FastCDR-1.0.8-Linux && ./configure --libdir=/usr/lib && make -j$cpucores && sudo make install)

(cd eProsima\_FastRTPS-1.7.1-Linux && ./configure --libdir=/usr/lib && make -j$cpucores && sudo make install)

rm -rf requiredcomponents eprosima\_fastrtps-1-7-1-linux.tar.gz

popd

fi

# jMAVSim simulator dependencies

echo "Installing jMAVSim simulator dependencies"

sudo apt-get install ant openjdk-8-jdk openjdk-8-jre -y

或是直接复制使用网盘附件中的同名文件ubuntu\_sim\_common\_deps.sh。

然后为该脚本添加权限：

sudo chmod +x ubuntu\_sim\_common\_deps.sh

打开一个新的终端，执行

sudo usermod -a -G dialout $USER

之后重启操作系统。重启后运行脚本：

source ubuntu\_sim\_common\_deps.sh

等待终端中出现done，表示toolchain安装完成。

接下来进行PX4 Firmware的下载和子模组配置。官方教程如下：

mkdir PX4

cd PX4

git clone https://github.com/PX4/Firmware.git

cd Firmware

git checkout v1.9.2

git submodule update --init --recursive

如果git速度较慢，也可以直接从网盘附件中下载课程组配置好的PX4固件，并重新进行编译。对于这种方法，需要先对固件中的脚本添加权限：

cd ~/PX4/Firmware/Tools

chmod +x \*.sh

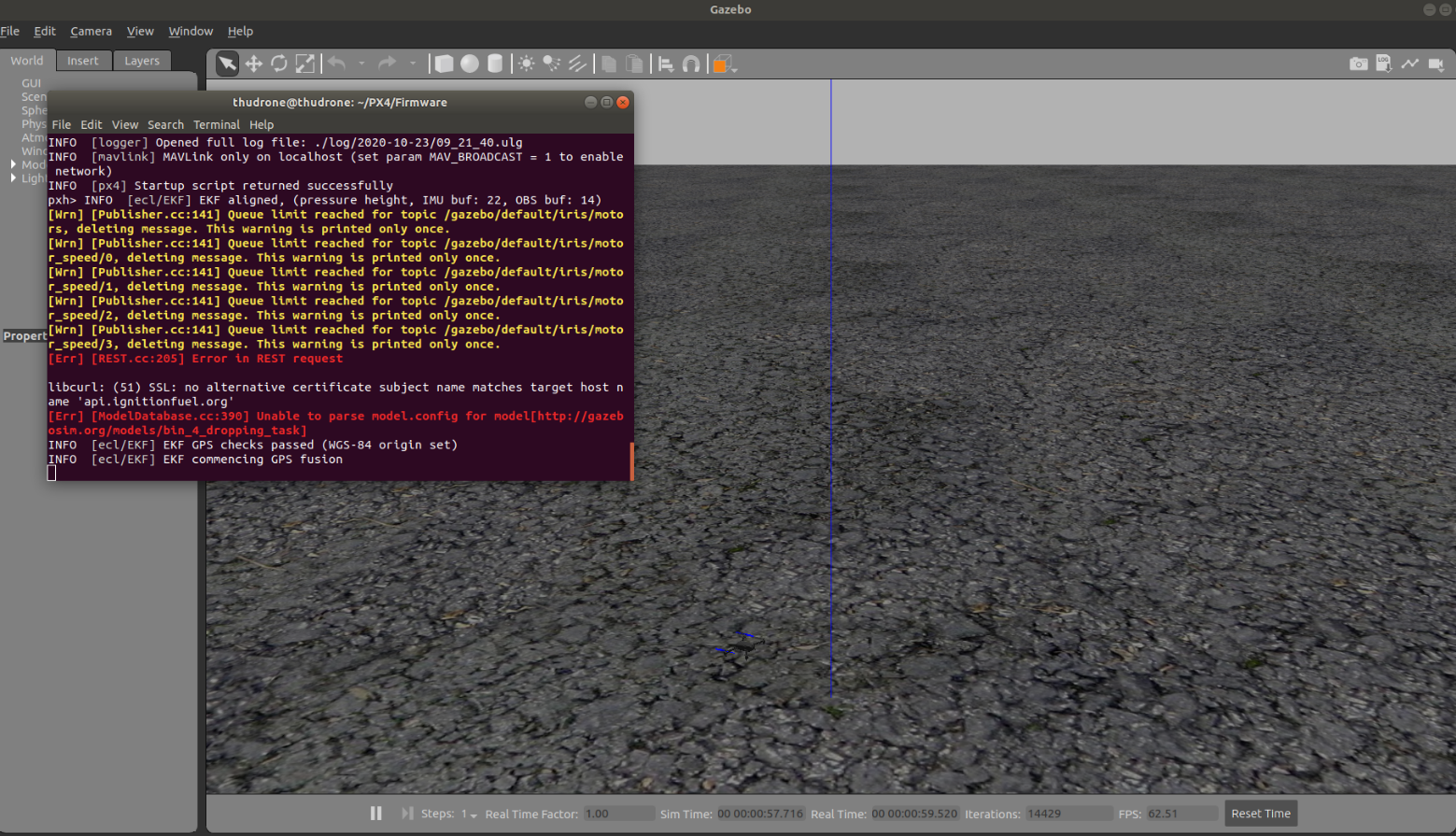
无论通过何种方式配置PX4固件，都需要如下进行编译：

cd ~/PX4/Firmware

make px4\_sitl\_default

make px4\_sitl\_default gazebo

上述最后一步编译完成时，会跳出一个Gazebo图形界面（如下所示），这说明PX4固件已经编译完成，可以在终端中Ctrl-C停止编译了。



最后在~/.bashrc后添加如下语句，以添加PX4环境变量：

export FIRMWARE\_DIR={填写PX4固件路径}

source $FIRMWARE\_DIR/Tools/setup\_gazebo.bash $FIRMWARE\_DIR $FIRMWARE\_DIR/build/px4\_sitl\_default

export ROS\_PACKAGE\_PATH=$ROS\_PACKAGE\_PATH:$FIRMWARE\_DIR

export ROS\_PACKAGE\_PATH=$ROS\_PACKAGE\_PATH:$FIRMWARE\_DIR/Tools/sitl\_gazebo

**3. 安装MAVLink和MAVROS**

首先我们用apt安装MAVLink和MAVROS的基础组件：

sudo apt install ros-melodic-mavros ros-melodic-mavros-extras

完成后进行GeographicLib的配置：

wget https://raw.githubusercontent.com/mavlink/mavros/master/mavros/scripts/install\_geographiclib\_datasets.sh

sudo bash ./install\_geographiclib\_datasets.sh

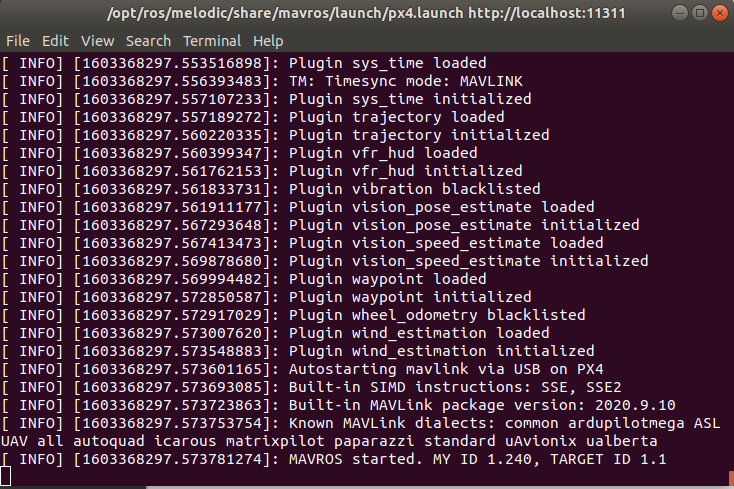
如果因为网络不好或部分下载源损坏等报错，可以直接将网盘附件中提供的GeographicLib文件夹下载后移至/usr/share即可：

sudo mv GeographicLib /usr/share

完成上述步骤后，可以执行命令：

roslaunch mavros px4.launch fcu\_url:="udp://:14540@127.0.0.1:14557"

如果没有报错，说明安装成功。（信息如下）



**4. 运行仿真例程**

在运行仿真例程前，需要先安装scipy：

pip install scipy -i https://pypi.tuna.tsinghua.edu.cn/simple/

附件中的catkin\_sim文件夹是仿真例程（及其可能依赖项）所在的ROS工作空间，该工作空间中的uav\_sim是仿真例程所在的ROS package。将catkin\_sim文件夹移至主目录下后，给文件夹中的Python脚本添加权限，并进行编译：

cd ~/catkin\_sim/src/uav\_sim/scripts

chmod +x \*.py

cd ~/catkin\_sim

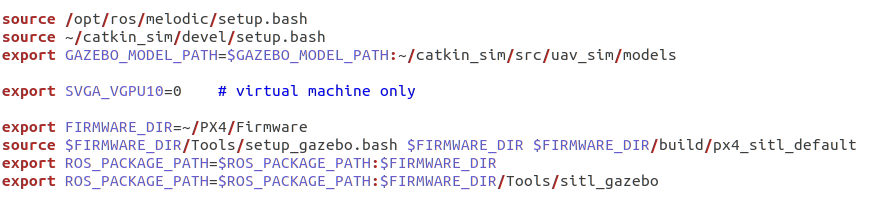
catkin\_make

将该工作空间相关变量添加到~/.bashrc中，即将以下语句

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

export GAZEBO\_MODEL\_PATH=$GAZEBO\_MODEL\_PATH:~/catkin\_sim/src/uav\_sim/models

添加至~/.bashrc文件设置PX4环境变量的语句之前。完成后，~/.bashrc的最后部分大致如下：



完成上述步骤后，即可运行仿真例程。打开一个新终端，运行命令：

roslaunch uav\_sim tello.launch

向/tello/cmd\_string发布tello格式控制信息即可控制仿真器内无人机的行为。例如，在另一个终端输入以下命令将使无人机起飞：

rostopic pub /tello/cmd\_string std\_msgs/String "data: 'takeoff'" -1