ROS Python使用说明

1.示例程序下载

点击下载示例程序 wit.rar（35 KB）

2.环境安装

Robot Operating System (ROS) 是一个得到广泛应用机器人系统的软件框架，它包含了一系列的软件库和工具用于构建机器人应用。从驱动到最先进的算法，以及强大的开发者工具，ROS 包含了开发一个机器人项目所需要的所有东西。且它们都是开源的。

ROS 虽然名为机器人操作系统，但它与我们一般概念中的操作系统，如 Windows，Linux，iOS 和 Android 这些。Windows，Linux，iOS 和 Android 这些操作系统为我们管理计算机的物理硬件资源，如 CPU、内存、磁盘、网络及外设，提供如进程、线程和文件这样的抽象，并提供如读文件、写文件、创建进程、创建线程及启动线程这样的操作。ROS 所工作的层级并没有这么低，它基于一般概念中的操作系统来运行，官方推荐基于 Ubuntu Linux 运行，并在 Ubuntu Linux 操作系统提供的抽象和操作的基础之上，提供了更高层的抽象，如节点、服务、消息、主题等，以及更高层的操作，如主题的发布、主题的订阅、服务的查询与连接等操作。同时 ROS 还提供开发机器人项目所需的工具和功能库。

ROS 发行版是一个版本标识的 ROS 包集合，这些与 Linux 发行版（如 Ubuntu）类似。ROS 发行版的目的是让开发者可以基于一个相对稳定的代码库来工作，直到他们可以平稳地向前演进。一旦发行版发布，官方就会限制对其的改动，而仅仅提供对于核心包的 bug fixes 和非破坏性的增强。

当前（2018-01-28） ROS 系统已经发布了多个版本。ROS 最新的一些版本如下：

ROS 系统版本	时间发布	支持时间
ROS Lunar Loggerhead	May 23rd, 2017	May, 2019
ROS Kinetic Kame	May 23rd, 2016	LTS，April, 2021 (Xenial EOL)
ROS Jade Turtle	May 23rd, 2015	May, 2017
ROS Indigo Igloo	July 22nd, 2014	LTS，April, 2019(Trusty EOL)
ROS Hydro Medusa	September 4th, 2013	May, 2015

ROS 基本上保持每年一个新版本，每两年一个长期发行版的发布节奏。关于 ROS 版本发布的更多内容，如更多的发行版的介绍，发布的计划等，可以参考 ROS 官方站点的 Distributions 主页。

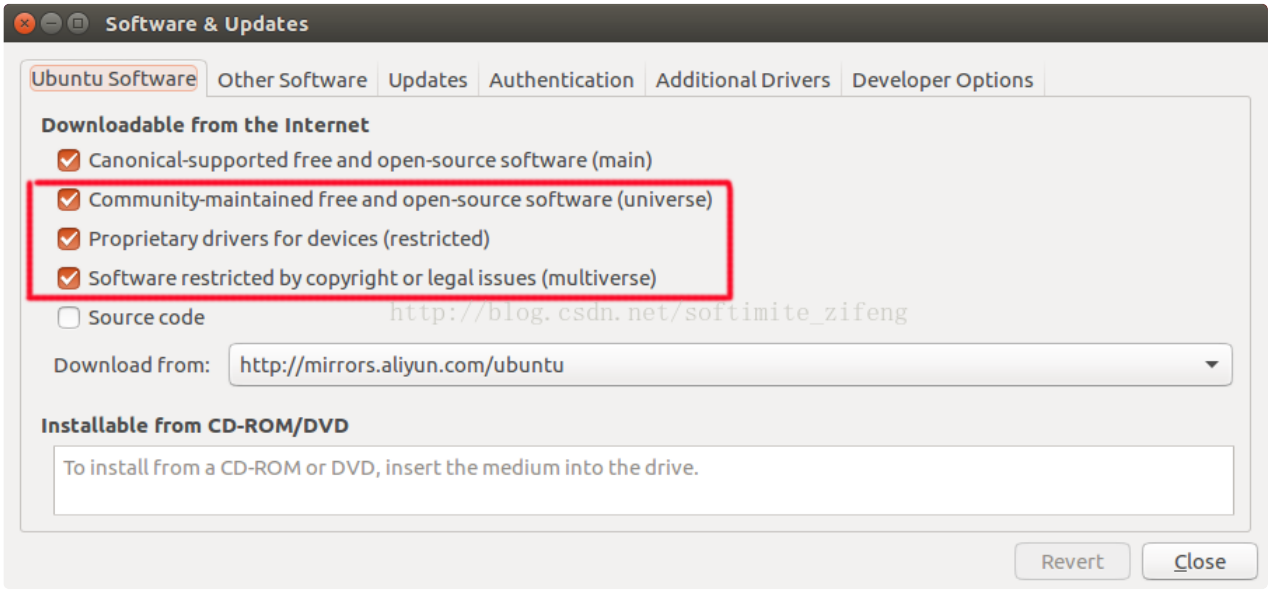
目前官方推荐使用最近的一个长期支持版本，即 ROS Kinetic Kame，求新的同时兼顾稳定性无疑应该采用这一版本，如果想要尝试最新的功能特性则可以使用最新的发行版 ROS Lunar Loggerhead。

ROS 的安装步骤如下：

ROS kinetic官网：http://wiki.ros.org/kinetic/Installation/Ubuntu <http://>

2.1 配置资源库

"restricted"，"universe"和"multiverse"。一般情况是不用配置的，参考下图红色框部分。如果没有配置，可以参考Ubuntu官网。



2.2 安装

1.设置安装源

```
1  sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.l
```

2.设置key

```
1  sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116
```

3.安装

```
1  sudo apt-get update
2  sudo apt-get install ros-kinetic-desktop-full
```

独立的包安装：可以安装一个特定的 ROS 包（用实际的包名来替换下面的命令中的 “PACKAGE”）。
sudo apt-get install ros-kinetic-PACKAGE
如：sudo apt-get install ros-kinetic-slam-gmapping
要找到可用的包，可以使用：apt-cache search ros-kinetic

2.3 初始化

```
sudo rosdep init  （安装一次后只能运行一次，若重新安装选择跳过或卸载）
rosdep update结果：
Add distro "groovy"
Add distro "hydro"
Add distro "indigo"
Add distro "jade"
Add distro "kinetic"
Add distro "lunar"
Add distro "melodic"
updated cache in /home/ubuntu1604/.ros/rosdep/sources.cache
```

若出现time out 或错误，重试几遍

2.4 配置环境变量

```
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc打开另一个终端：
source ~/.bashrc
```

如果安装了多个 ROS 发行版，则 ~/.bashrc 必须只 source 当前正在使用的那一版的 setup.bash。

如果你只想要修改当前 shell 的环境，则输入如下的命令来替换上面的命令：source /opt/ros/kinetic/setup.bash

2.5 安装rosinstall

构建包所需的依赖

```
sudo apt-get install python-rosinstall
```

到这一步，应该已经安装好了运行核心 ROS 包的所有东西。要创建和管理你自己的 ROS workspace，还有单独发布的许多的工具。比如，rosinstall 是一个常用的命令行工具，使你可以通过一个命令为 ROS 包简单地下载许多源码树。
要安装这个工具及其它的依赖以构建 ROS 包，则运行：

```
sudo apt-get install python-rosinstall python-rosinstall-generator python-wstool build-essential
```

完成完整的 ROS 安装之后，可以对安装做一个简单的测试。可以通过 roscore 和 turtlesim 来做测试。

2.6 测试

```
1）打开Termial，输入以下命令，初始化ROS环境：
roscore
2）打开新的Termial，输入以下命令，弹出一个小乌龟窗口：
rosrun turtlesim turtlesim_node
3）打开新的Termial，输入以下命令，可以在Termial中通过方向键控制小乌龟的移动：
rosrun turtlesim turtle_teleop_key
```

3.IMU软件包使用

3.1 安装 ROS IMU 依赖项

请在终端运行对应的命令

如果你使用的是 ubuntu 16.04，ROS kinetic，python2 :

```
sudo apt-get install ros-kinetic-imu-tools ros-kinetic-rviz-imu-plugin
sudo apt-get install python-visual
```

如果你使用的是 ubuntu 18.04，ROS Melodic，python2 :

```
sudo apt-get install ros-melodic-imu-tools ros-melodic-rviz-imu-plugin
```

如果你使用的是 ubuntu 20.04，ROS Noetic，python3 :

```
sudo apt-get install ros-noetic-imu-tools ros-noetic-rviz-imu-plugin
pip3 install pyserial
```

3.2 建立工作空间

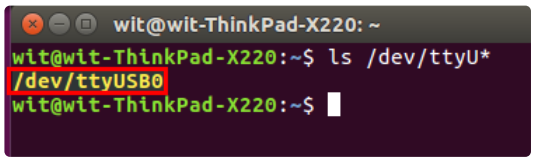
下载示例程序，将文件放到home根目录，右击提取到此处。
打开命令终端，运行下面指令：

```
cd ~/wit/wit_ros_ws/
catkin_make
cd ~/wit/wit_ros_ws/src/wit_ros_imu/scripts/
sudo chmod 777 *.py
echo "source ~/wit/wit_ros_ws/devel/setup.sh" >> ~/.bashrc
source ~/.bashrc
```

3.3 ROS 驱动和可视化

以 ubuntu16.04，JY901S，python2.7 为例

1. 查看USB端口号。先不要插 IMU 的 USB ，在终端输入 `ls /dev/ttyUSB*` 来检测一下，然后在将 USB 插入电脑，再在终端输入 `ls /dev/ttyUSB*` 来检测一下，多出来的 ttyUSB 设备就是 IMU 的串口。



2. 修改参数配置。需要修改的参数包括设备类型，USB端口号和波特率。进入脚本目录~/wit/wit_ros_imu/src/launch，修改对应的 launch 文件中的配置参数。设备类型如果是modbus协议的就填modbus，使用wit标准协议的填normal。设备号/dev/ttyUSB0（脚本默认用的/dev/ttyUSB0）为你电脑识别出来的数字。波特率根据实际使用设定，JY6x系列模块默认波特率为115200,其他模块为9600,如果用户通过上位机修改了波特率，需要对应修改成修改后的波特率。

display_and_imu.launch (~/wit/wit_ros_ws/src/launch) - gedit

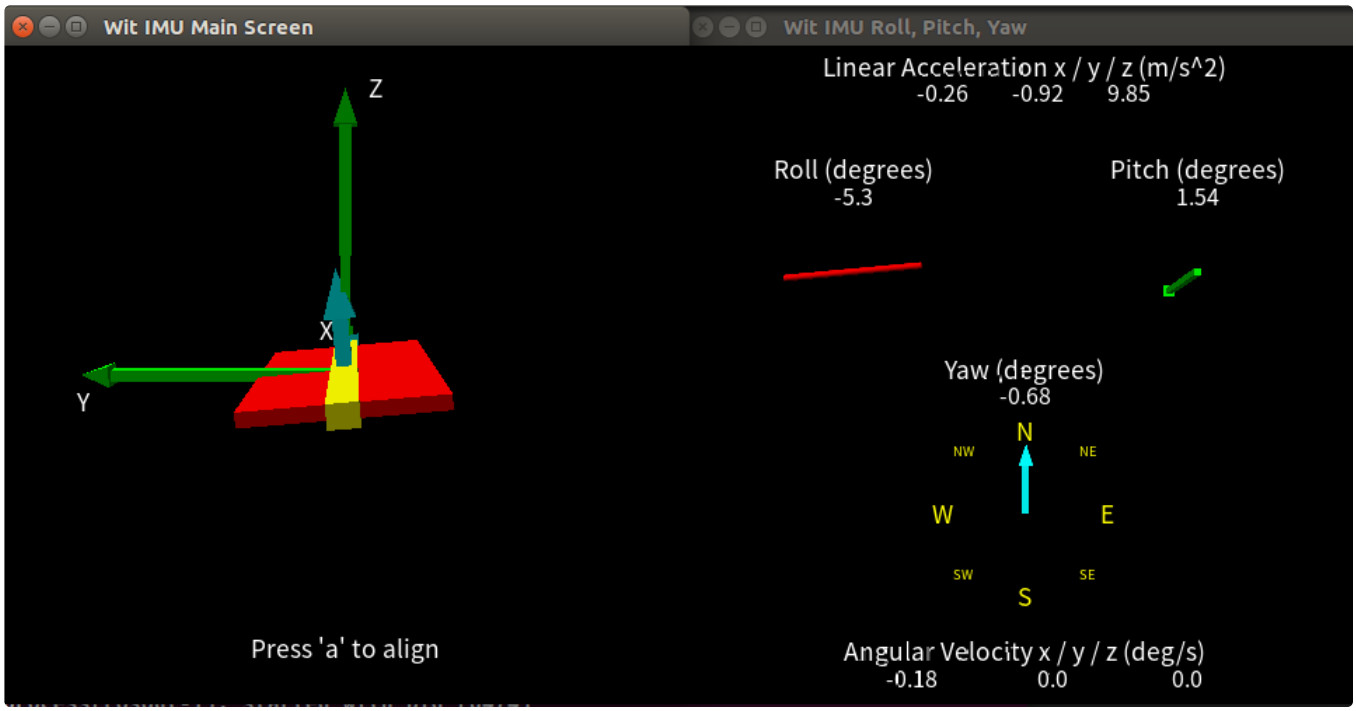
打开(O)

```
<!--imu 和 3D 模型同时打开-->

<launch>

    <!-- imu型号，默认 normal -->
    <arg name="type" default="normal" doc="type [normal, modbus]"/>

    <!-- imu 对应 python 文件 -->
    <node pkg="wit_ros_imu" type="wit_$(arg type)_ros.py" name="imu" output="screen">
        <param name="port"              type = "str"    value="/dev/ttyUSB0"/>
        <param name="baud"              type = "int"    value="9600"/>
    </node>

    <!-- 打开 3d 模型 -->
    <node pkg="wit_ros_imu" type="display_3D_visualization.py" name="display_3D_visualizat
    </node>

</launch>
```

3. 给对应的串口管理员权限，在终端输入： sudo chmod 777 /dev/ttyUSB0，提示你输入管理员密码，输入密码后回车即可。注意每次重新插入USB口都需要重新给串口赋管理员权限。
4. 打开终端，运行 imu 驱动， `imu_type:=*` * 为对应的型号，可选择有[normal, modbus]

Plain Text

```
roslaunch wit_ros_imu visual_and_.launch
```

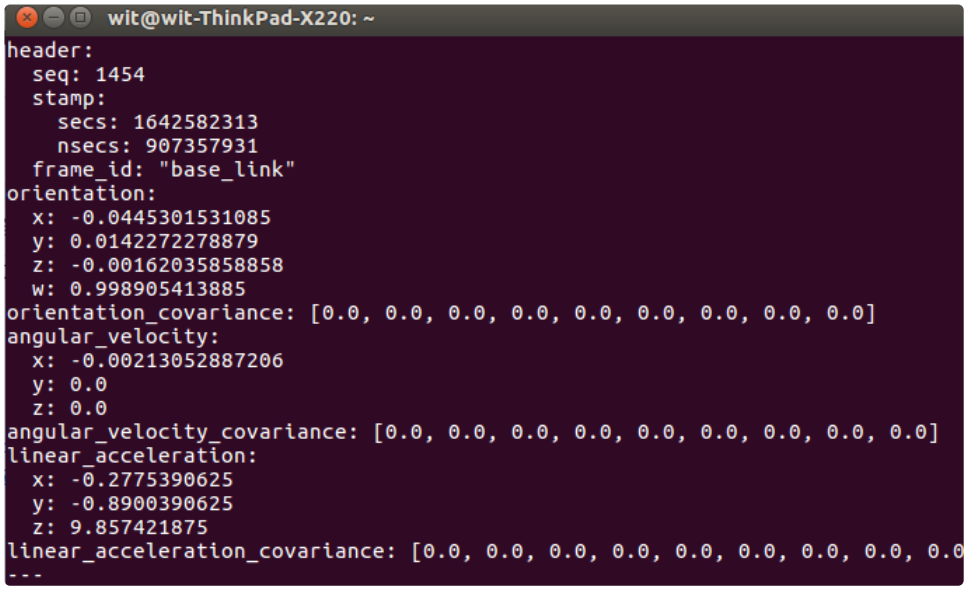


打开两个新终端输入分别输入下面两行命令

Plain Text

```
rostopic echo /wit/imu
rostopic echo /wit/mag
```

如下图，驱动运行成功后和 `rostop echo` 输出的信息



4. 同理，如需要运行其他 launch 文件，需要先确保 launch 文件中的 /dev/ttyUSB 设备修改对。
5. 相关文件说明
   - display_and_imu.launch，打开打开 IMU 驱动节点和用 visual 编写的可视化模型。（仅支持 ubuntu 16.04）
   - wit_imu.launch，打开用 IMU 驱动节点。
   - rviz_and_imu.launch，打开 IMU 驱动节点和 Rviz 可视化。

操作具体教程示例：

打开ROS系统，运行相关命令

- 按住ctrl+Alt+T 打开终端 （ 查看当前设备下所有的串口文件）

终端输入命令：

```
ls -l /dev/ttyUSB*
```

- 使用三合一或者六合一接上维特智能传感器之后插入到有ROS系统电脑下的USB口
- 多出的串口即为我们需要的串口文件

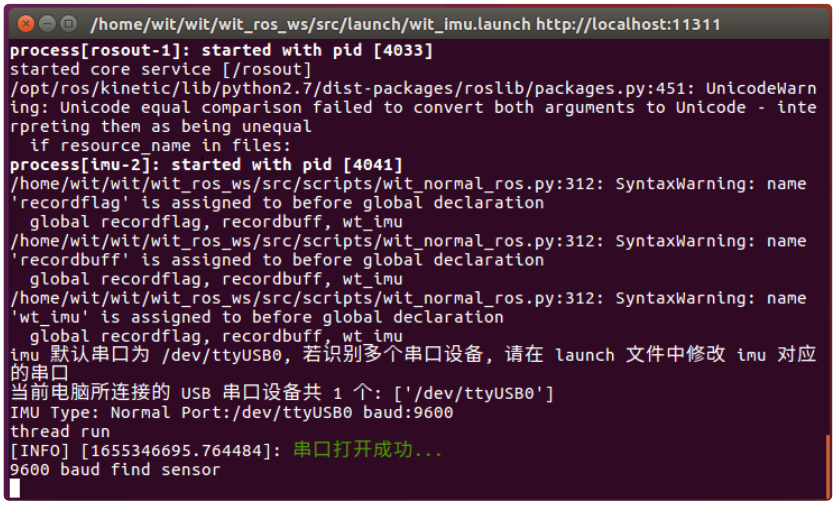
终端输入命令：

```
ls -l /dev/ttyUSB*
```

- 修改串口文件的权限以及执行终端命令

终端输入命令：

```
sudo chmod 777 /dev/ttyUSB0
cd wit/wit_ros_ws/src/launch
roslaunch wit_ros_imu wit_imu.launch
```

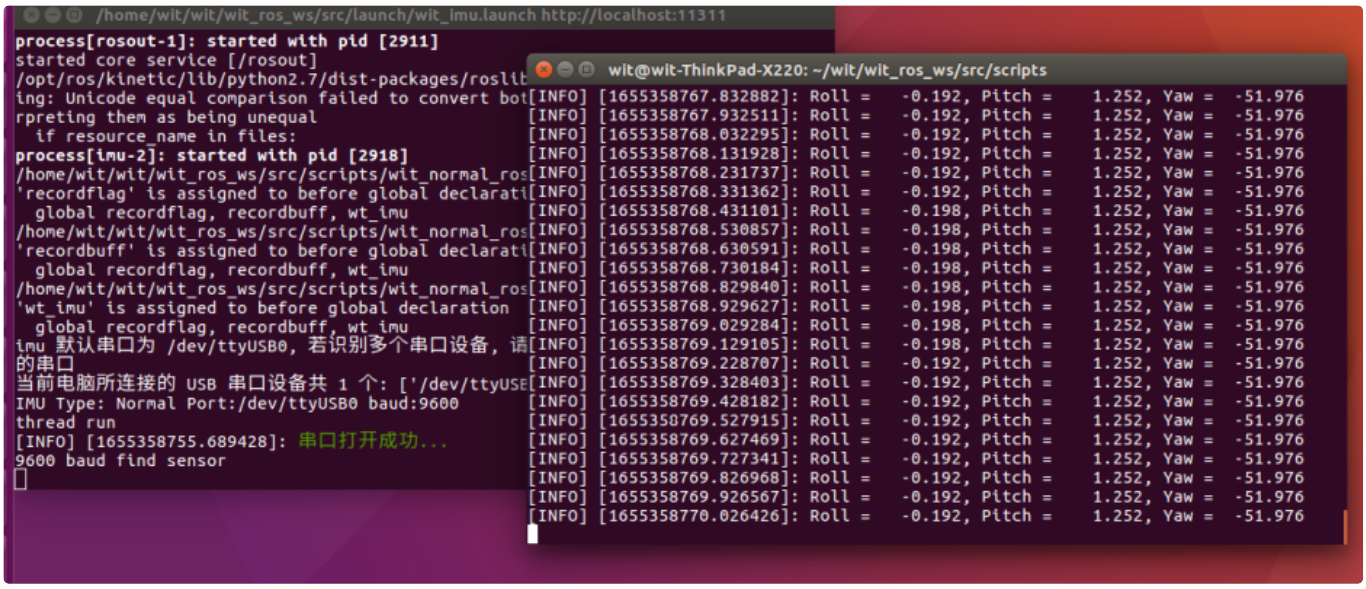


当串口成功打开时，会有提示信息，同时会自动检索当前传感器的波特率，并提示波特率的信息，如上图，成功检索到当前传感器的波特率为9600.

- 打开新的终端

终端输入命令：

```
cd wit/wit_ros_ws/src/scripts
python get_imu_rpy.py
```



此时该终端实时打印传感器回传的有关参数数据

- 打开新的终端

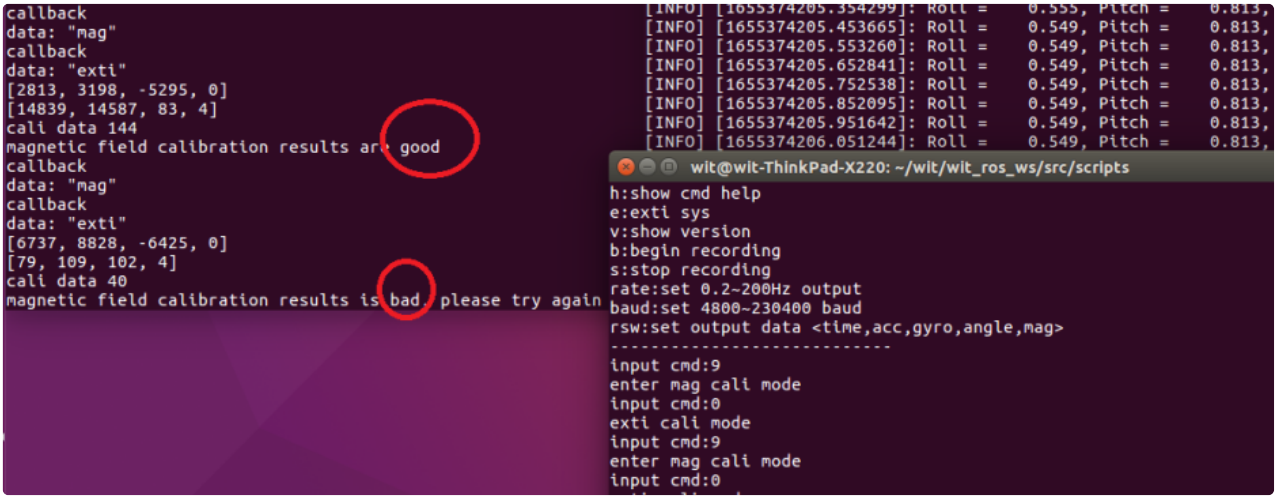
终端输入命令：

```
cd wit/wit_ros_ws/src/scripts
python wit_imu_ctrl.py
```

校准功能

在在input cmd下输入数字9，即可进入校准状态

将传感器水平放置缓慢旋转两圈后，终端输入数字0即可结束校准



当我们校准之后，提示信息为very good 或者good 说明校准结果良好；
当出现 bad 的提示时，说明校准结果不好，建议重新校准。

设置回传速率

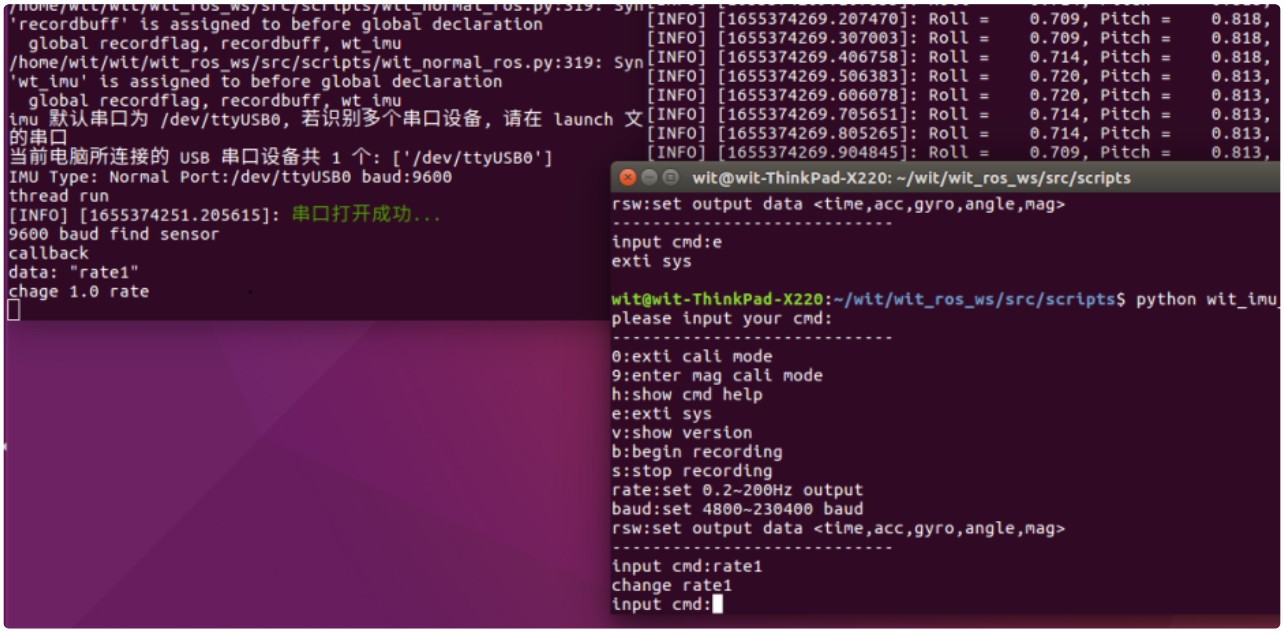
在input cmd下输入：

```
rate1
```

即可将回传速率设置为1Hz
此时第二个终端会每1s回传传感器检测到的数据

- 同理设置为200Hz，即为rate200

可设置命令：rate0.2、rate0.5、rate1、rate2、rate5、rate10、rate20、rate50、rate100、rate200



设置波特率

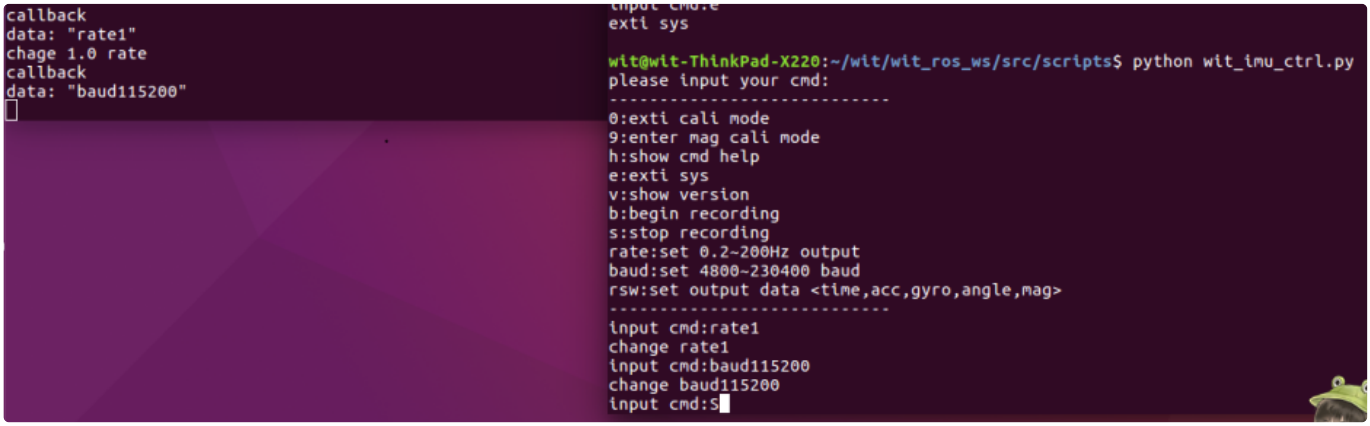
在input cmd下输入：

```
baud115200
```

即可将波特率设置为115200

- 同理波特率设置为9600，即为baud9600.

可设置命令：baud4800、baud9600、baud19200、baud38400、baud57600、baud115200、baud230400



提示：如何验证波特率修改成功

- 修改波特率成功之后第二个终端的数据是在实时刷新的
- 可以借助串口调试助手验证

记录与解析数据

以回传速率200Hz、波特率115200为例：

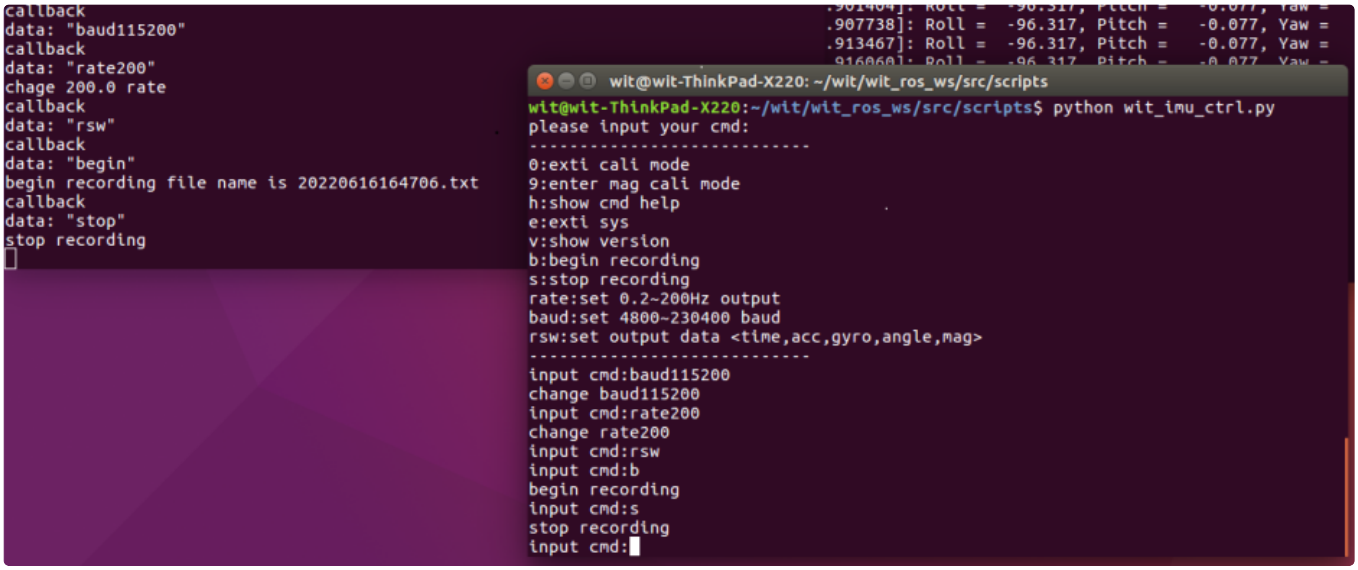
- 记录数据

  先将回传速率和波特率分别设置为200Hz、115200

  在input cmd下输入：rsw　　设置传感器回传的记录数据有片上时间、加速度、角速度、角度、磁场

  在input cmd下输入：b　　即可开始记录原始数据

  在input cmd下输入：s　　即可结束记录



记录的文件以当前系统年月日时分秒命名，存放的路径时： ~/.ros/

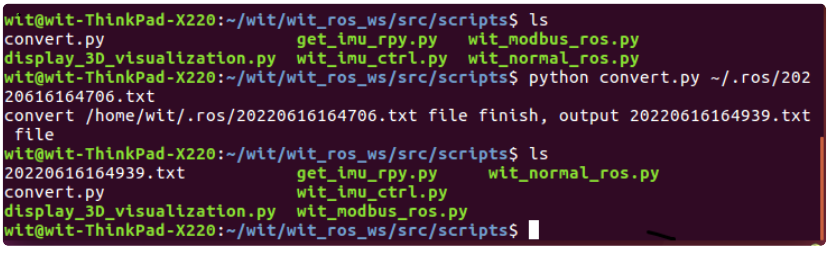
并且第一个终端会显示记录的文件名，如上图：20220616164706.txt

- 解析数据

终端输入命令：

Plain Text

```
cd wit/wit_ros_ws/src/scripts
python convert.py ~/.ros/20220616164706.txt
```



当解析数据成功时，会显示成功解析后的文件名。

终端输入命令：ls

可查看到当前路径下新增了解析后的数据文本

用文本编辑器即可打开文本

```
t-ThinkPad-X220: ~/wit/wit_ros_ws/src/scripts
AngleX(deg)     AngleY(deg)     AngleZ(deg)      hx      hy      hz      Chip-Time       ax(g)   ay(g)   az(g)   wx(deg/
 wz(deg/s)
-94.779 -0.555  92.027  -39     -245    -31     2000-00-00 00:34:45.225 0.009   -0.995  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -38     -254    -33     2000-00-00 00:34:45.230 0.009   -0.995  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -37     -261    -34     2000-00-00 00:34:45.235 0.009   -0.995  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -37     -267    -35     2000-00-00 00:34:45.240 0.009   -0.995  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -37     -273    -34     2000-00-00 00:34:45.245 0.009   -0.996  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -40     -279    -33     2000-00-00 00:34:45.250 0.009   -0.996  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -43     -284    -31     2000-00-00 00:34:45.255 0.009   -0.996  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -45     -289    -30     2000-00-00 00:34:45.260 0.009   -0.996  -0.084  0.000   0.000   0.000
-94.779 -0.555  92.027  -47     -293    -29     2000-00-00 00:34:45.265 0.009   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.555  92.027  -47     -260    -28     2000-00-00 00:34:45.270 0.009   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.555  92.027  -47     -231    -28     2000-00-00 00:34:45.275 0.009   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -47     -241    -30     2000-00-00 00:34:45.280 0.009   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -48     -249    -33     2000-00-00 00:34:45.285 0.010   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -48     -255    -35     2000-00-00 00:34:45.290 0.010   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -48     -260    -37     2000-00-00 00:34:45.295 0.010   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -47     -265    -37     2000-00-00 00:34:45.300 0.010   -0.995  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -45     -269    -37     2000-00-00 00:34:45.305 0.011   -0.997  -0.083  0.000   0.000   0.000
-94.779 -0.549  92.027  -45     -274    -35     2000-00-00 00:34:45.310 0.011   -0.997  -0.083  0.000   0.000   0.000
```

可以看到数据每5ms回传一次数据，说明回传速率达到200Hz/s

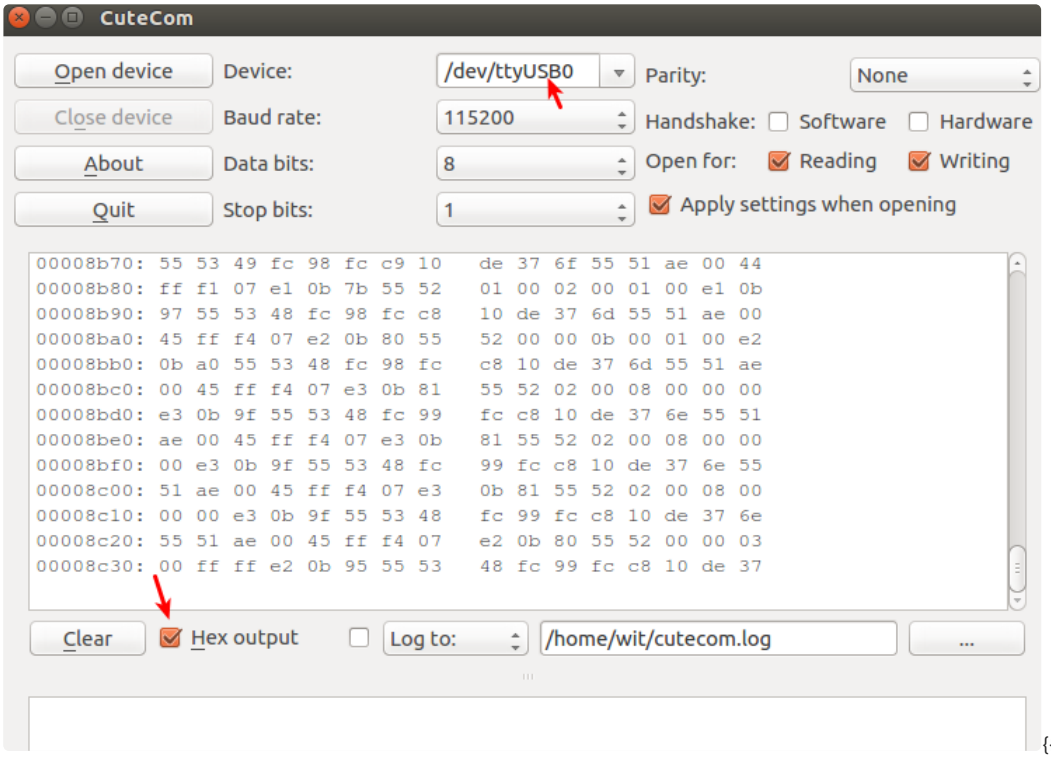
4. 串口助手测试通讯

以 ubuntu16.04 为例，波特率 9600

1. 安装 cutecom（你也可以安装其他的串口助手进行调试）。

```
apt-get install cutecom -y
```

2. 插入串口模块，需要通过命令ls /dev/ttyU*查看新加入的串口模块的端口号，然后对其赋予读写权限，以/dev/ttyUSB0为例，输入指令：sudo chmod 777 /dev/ttyUSB0，然后根据提示输入管理员密码。注意每次新插入USB模块都需要重新赋读写权限。
3. 安装成功后在终端输入 cutecom，打开串口助手，然后进行一些设置，注意从下拉框中选择的串口号是不对的，需要根据上面第二步中获取到的串口号进行修改，比如/dev/ttyUSB0，如图中所示。



{{{width="100%" height="auto"}}}

4. 然后我们点击 `open device` ，此时下面的空白面板会有 imu 的数据打印。
5. 我们可以等待 imu 的数据打印一会儿，然后点击 `close device` 来查看。
6. 如果可以找到 `55 51` 、 `55 52` 、 `55 53` 开头的信息，那么模块发送的数据是没有问题的，如果有数据但没有找到正确包头的数据，则需要核对一下波特率设置，切换到正确的波特率就可以显示了。