

XQ-4 Pro

ROS is not difficult any more!

XQ-mini



用户手册

(综合版)



蓝鲸智能机器人(深圳)有限公司

目录

一、开始使用.....	3
1.设置网络.....	3
2.产品组装.....	3
3.状态检查.....	3
4.远程遥控.....	4
5.视频传输.....	5
6.软件整体结构和说明.....	5
7.ROS 入门手册.....	5
二、小强 ROS 机器人教程.....	6
教程(1)___基础操作介绍.....	6
1.1 配置小强网络.....	7
1.2 本地遥控端配置.....	8
1.3 配置本地 ubuntu 系统.....	9
C.安装了小强系统镜像的用户，请关闭开机启动项，避免与小强冲突.....	9
教程(2)___蓝鲸智能开源软件仓库的使用和 ROS 开机启动任务的配置.....	13
教程(3)___在 rviz 中显示小强机器人模型.....	16
教程(4)___惯性导航自主移动测试.....	25
教程(5)___小强遥控图传 app 安卓版.....	28
教程(6)___小强遥控图传 windows 客户端.....	31
教程(7)___使用 ps3 手柄控制小强.....	33
教程(8)___kinect1 代 ROS 驱动测试与安装.....	37
教程(9)___使用 rostopic 控制 kinect 的俯仰角度.....	41
教程(10)___使用 kinect 进行自主移动避障.....	43
教程(11)___kinect 跟随包 turtlebot_follower.....	48
教程(12)___ROS 显示 kinect2 代的点云.....	50
教程(13)___rplidar 二代激光雷达的使用暨利用 udev 给小车增加串口设备.....	51
教程(14)___在 gmapping 下使用激光雷达 rplidar a2 进行建图.....	54
教程(15)___AMCL 导航测试.....	57
教程(16)___大范围激光雷达 slam 与实时回路闭合测试.....	61
教程(17)___利用 ORB_SLAM2 建立环境三维模型.....	64
教程(18)___利用 DSO_SLAM 建立环境三维模型.....	69
教程(19)___NLinepatrol_planner 的简单使用.....	71
教程(20)___获取小车视觉里程计并在 rviz 中显示小车轨迹.....	74
教程(21)___获取 usb 摄像头 30fps 的 1080p 图像流及 120fps 的 VGA 分辨率图像流....	77
教程(22)___操作 6 自由度机械臂.....	80
教程(23)___ROS 入门手册.....	81
三、维护.....	82
1 充电.....	82
2 车轮松动打滑.....	82
3 小强底盘固件的自动更新升级方法.....	83
3.1 升级固件包软件.....	83

3.2 开始升级底盘固件	83
3.3 升级失败处理办法	85
4 升级底盘 ros 驱动包 xqserial_server	86
5 重新校准小车底盘 IMU	87
6 小强系统镜像	88
6.1 镜像使用方法	88
四、Ubuntu 设置静态 IP	96
五、视觉导航路径编辑器使用教程	101
六、小强的远程协助功能	108
七、小强 ROS 机器人障碍物识别演示	110
八、视觉导航在履带车中的运用	111
九、Google 激光雷达 slam 算法 Cartographer 的安装及 bag 包 demo 测试	112
十、原装和国产 ps3 手柄 ros 驱动程序	115
十一、升级软件包以支持小强图传遥控 app	118
十二、附件	119
1. 小车系统框架图	119
2. 电气布线图	120
3. 小强电脑与 stm32 底层通讯协议	121

一、开始使用

本章介绍平台的快速使用方法，如果您对 ROS 系统和 Ubuntu 系统不熟悉，请先阅读第二章的[\[教程一\]](#)。

1. 设置网络

首先将小强的主机连接上电脑显示器(小强 pro 用户请利用附赠的 HDMI 到 VGA 转接头进行连接, 小强 mini 用户请直接使用 vga)。小强的默认密码是 xiaoqiang, 请及时更改默认密码。进入系统设置好小强的 wifi 网络连接。推荐设置路由器使小强使用静态 ip,[\[静态 IP 设置方法\]](#), 方便以后连接。

2. 产品组装

小强的主要部分如下图所示



将电池平放在主机前方空余区域(电池靠两个小黑块堵住), 根据线标提示接上底盘电源线, 安装好电脑主机 wifi 天线, 摄像头和底层 USB 连接模块连接上主机的 USB 接口即可。

3. 状态检查

组装完成之后就可以开始使用小强了。打开小强主机开关, 等待主机蓝色灯亮起。小强左侧电源数据显示正常(电池电压合理使用范围是 10V 以上, 当电池电压小于 10V 时小强会自动关机)。

通过 ssh 远程连接，其中 xxx.xxx.xxx.xxx 是小强的 IP

```
ssh xiaoqiang@xxx.xxx.xxx.xxx
```

检查程序是否正常运行，执行以下指令

```
rostopic list
```

正常情况下会显示出当前的所有 ROS topic.

```
/ORB_SLAM/Camera  
/ORB_SLAM/Frame  
/camera_node/image_raw  
/orb_scale/scaleStatus  
/rosout  
/rosout_agg  
/system_monitor/report  
/tf  
/usb_cam/brightness  
/xqserial_server/Odom  
/xqserial_server/Power
```

如果没有正常显示 topic，可以尝试重启 service。

```
sudo service startup restart
```

查看系统状态

```
rostopic echo /system_monitor/report
```

如果正常，则显示如下

```
imageStatus: True  
odomStatus: True  
orbStartStatus: False  
orbInitStatus: False  
orbScaleStatus: False  
brightness: 0  
power: 12.34432
```

其中 imageStatus 表示摄像头是否工作正常。odomStatus 表示底层驱动时候工作正常。Orb 相关的三个变量是视觉导航相关的状态，可以不用管（如果对这方面感兴趣可以在论坛里进行交流）。brightness 是摄像头的亮度，power 是当前电池的电压值，如果无法读取则是 0.

4. 远程遥控

通过 ssh 进行连接

```
ssh xiaoqiang@xxx.xxx.xxx.xxx
```

启动遥控程序

```
roslaunch nav_test control.py
```

现在就可以通过方向键来控制小强的移动了。空格键是停止。Ctrl + C 退出程序。

5. 视频传输

小强的视频也是可以远程传输的。在本地打开一个终端输入

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:1131
```

然后在本地系统的/etc/hosts 文件内添加小强的 ip

```
192.168.X.X xiaoqiang-desktop
```

输入下述命令开始显示摄像头视频

```
roslaunch image_view image_view image:=/camera_node/image_raw _image_transport:=compressed
```

如果一切正常就能够看到当前小强的摄像头画面了。

6. 软件整体结构和说明

小强的软件构建于 ROS 之上。程序主要包含有底层驱动，导航算法，slam 算法。

对于机器人来说，软件是一个整体，各部分的依赖程度都比较高。为了方便管理，系统中的基础功能用一个 service 统一操作。也就是前文中提到的 startup service。这个 service 会启动底层的驱动软件包和摄像头。

startup 软件包位于/home/xiaoqiang/Documents/ros/src/startup。系统启动时会自动启动这个服务。如果想要更改这个服务的内容，可以修改这个软件包内的 launch 文件。具体的操作请[参照这里](#)。

系统的导航程序采用的是 ROS 自带的导航程序包。不过导航参数是根据小强自己的参数修改过的。导航参数对导航的性能表现影响很大，你也可以尝试自己修改参数以提高表现。

slam 算法采用的是 ORB_SLAM，这个算法目前还无法用于实际的生产环境，但是使用效果也很不错。

最后就是一些工具类软件了。比如控制小车移动的 nav_test control.py，显示系统状态的 system monitor。

7. ROS 入门手册

[Learning ROS for Robotics Programming - Second Edition.pdf](#)(如果点击无法下载，请联系技术支持)。这本教程很基础、很全面，虽然以 Hydro 版本为例，但是也完全兼容 jade 版本，代码实例中只需将书中的 Hydro 字符串替换成 jade 即可。请重点阅读本书的第二章和第三章。

二、小强 ROS 机器人教程

教程(1)——基础操作介绍

首先请您自行依据线标提示将小强接线连接好，完整结构如下两图所示，

[小强 pro 淘宝购买链接](#)

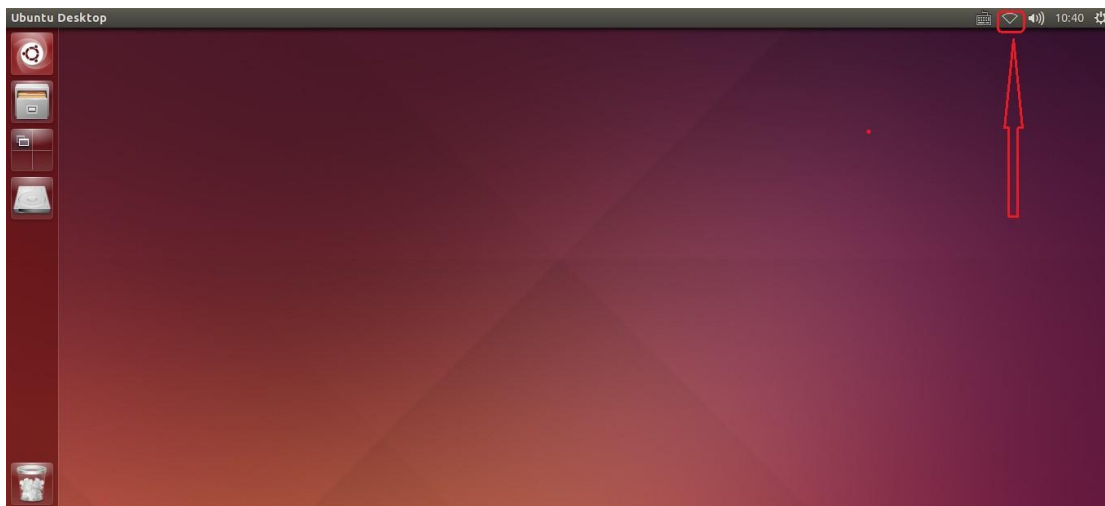
[小强 mini 淘宝购买连接](#)



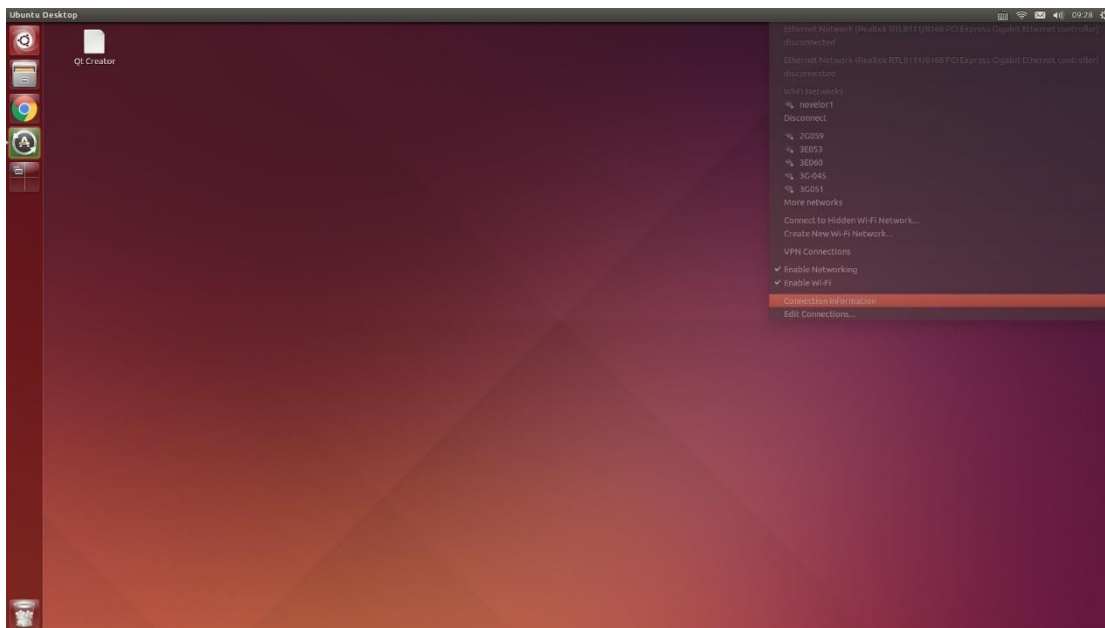
1.1 配置小强网络

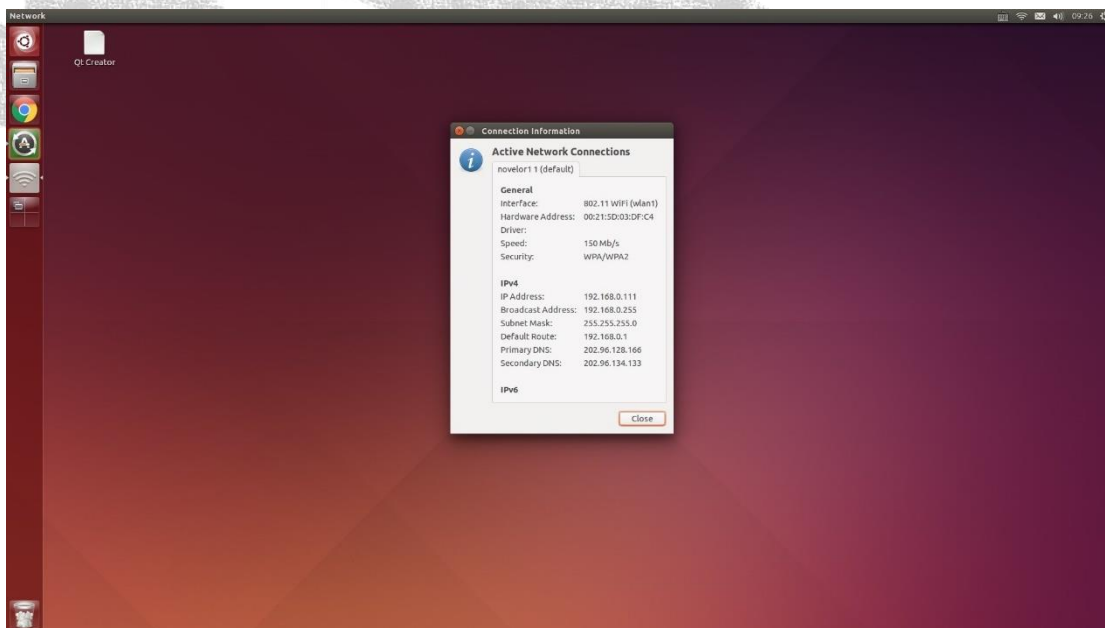
使用赠送的 **hdmi 转 vga** 插头（小强 mini 可以直接使用 **vga**），将显示器和键盘鼠标接入小强主机后开机，进入主机 **ubuntu** 系统。小强默认密码是 **xiaoqiang**

点击下图位置，选择需要接入的无线网络

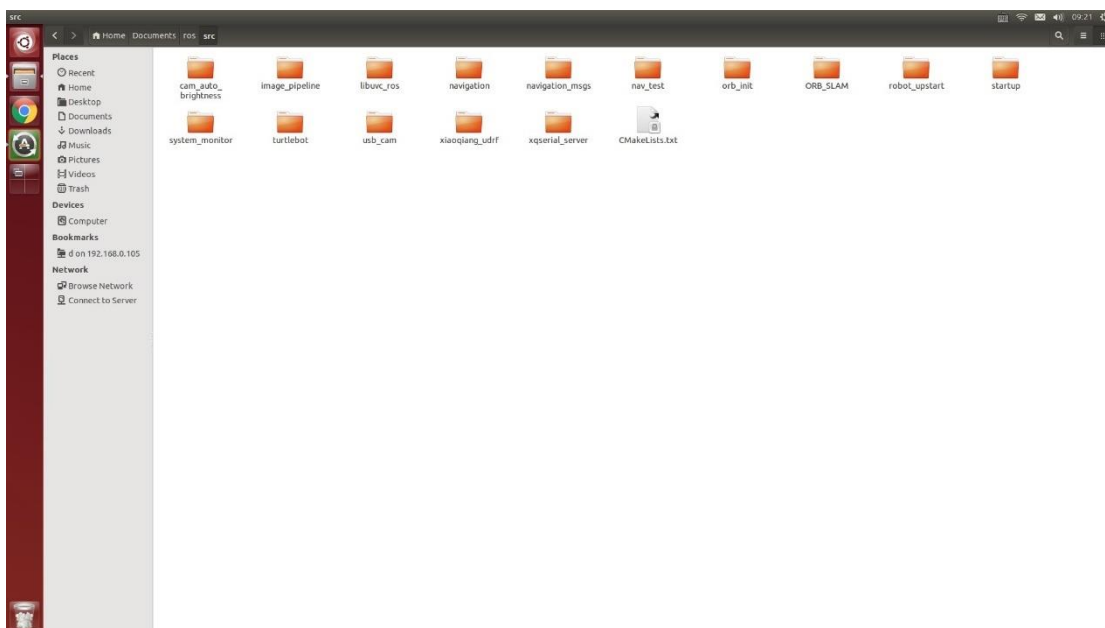


接入网络后，点击下图位置得到小强的实际 **ip** 地址，后续教程会频繁使用到这个 **ip** 信息





小强的 ROS 默认工作目录在这里



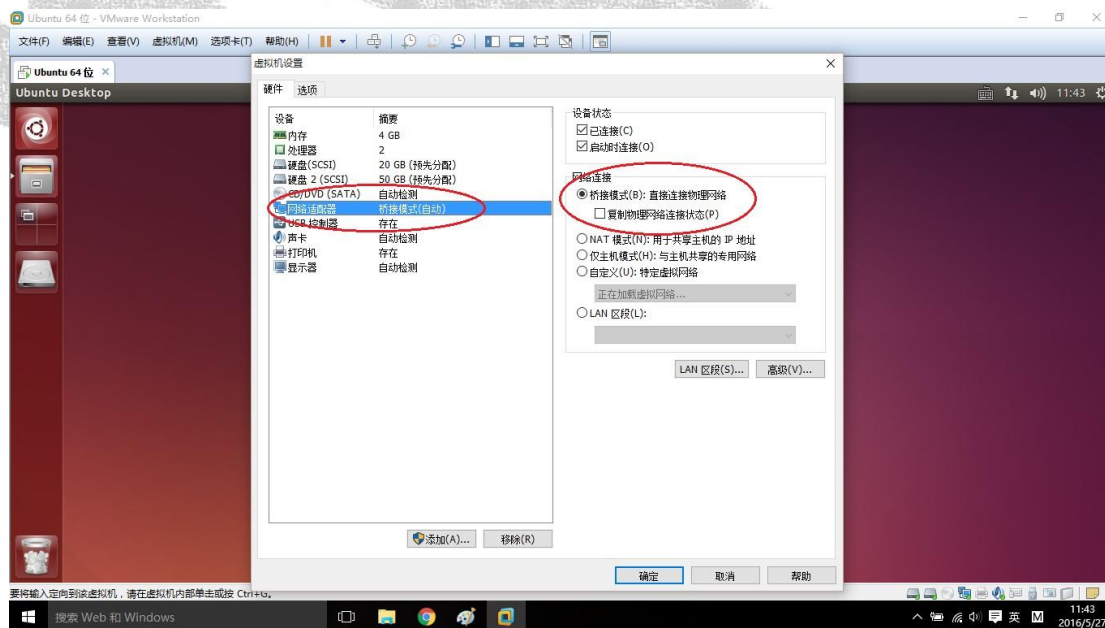
现在小强已经配置完成了，关机后拔掉键盘鼠标和显示器，下次小强直接开机就能使用

1.2 本地遥控端配置

本地遥控端最好是一台安装 `ubuntu14.04 64bit` 操作系统的 `x86` 主机，如果您只有 windows 平台，可以先安装 `vmware` 虚拟机 [[下载地址](#)]，然后在虚拟机里安装 `ubuntu`。

[[Ubuntu 小强镜像下载链接](#)]， [[镜像安装教程](#)]，推荐安装使用已经配置好 ROS 的小强镜像。

最后保证虚拟机必须和小强在同一个局域网下



1.3 配置本地 ubuntu 系统

A. 安装 SSH SCREEN, 如果安装了小强系统镜像请跳过这个操作

```
sudo apt-get install ssh screen
```

B. 安装 ROS JADE 版本, 如果安装了小强系统镜像请跳过这个操作

[官方教程](#), [中文教程](#)

C. 安装了小强系统镜像的用户, 请关闭开机启动项, 避免与小强冲突

```
sudo service startup stop
rosrun robot_upstart uninstall startup
```

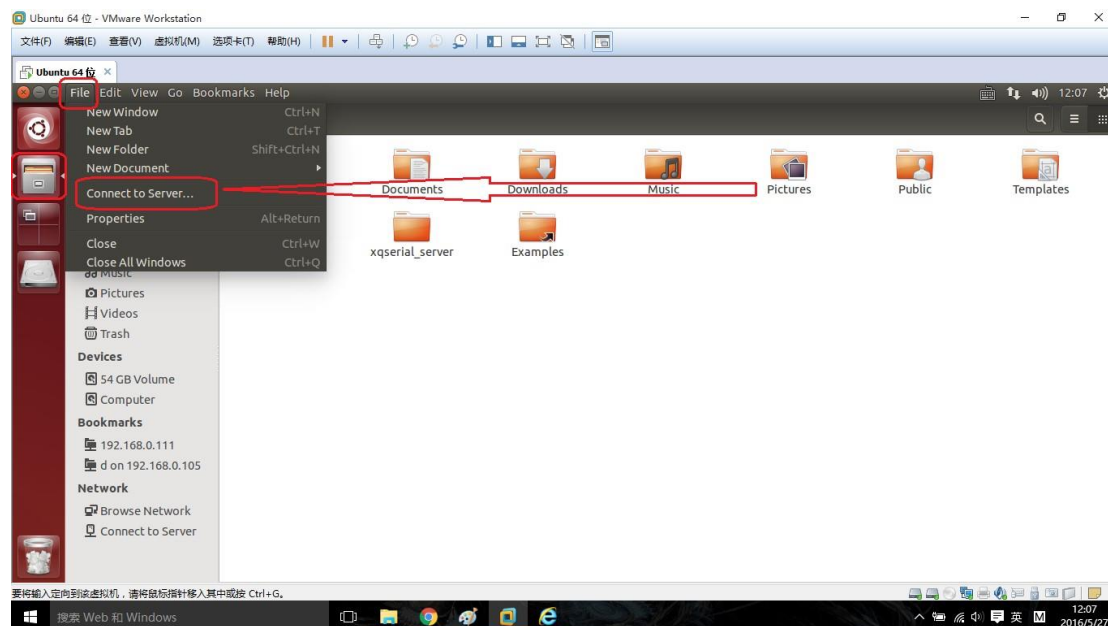
D. 安装了小强系统镜像的用户, 请根据小强型号 (PRO ? MINI), 更新 XIAOQIANG_UDRF 包

```
cd ~/Documents/ros/src/xiaoqiang_udrf
git stash
git pull
# 小强 pro 用户, 请切换到 master 分支
git checkout master
# 小强 mini 用户, 请切换到 mini 分支
git checkout mini
```

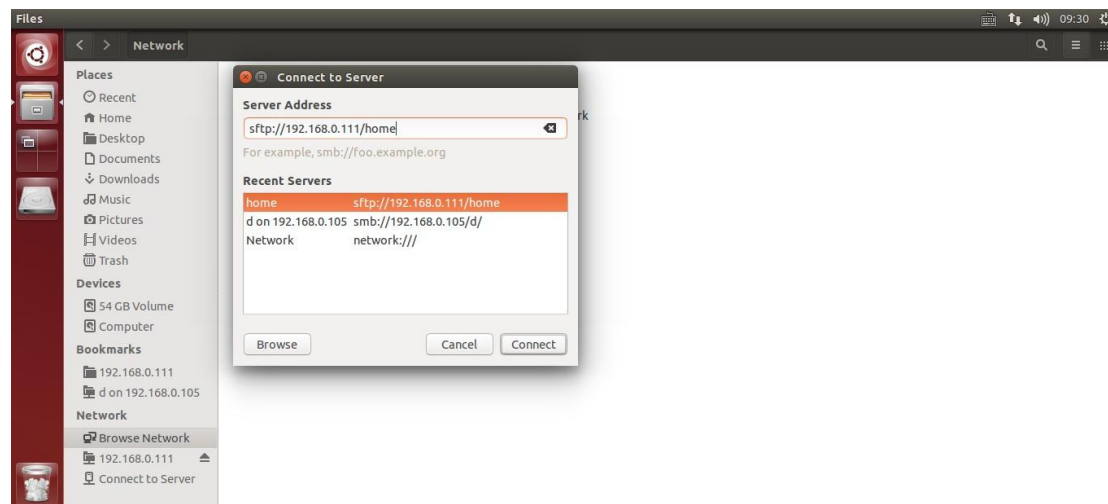
E. 添加远程目录

因为后续需要经常操作更改小强主机上的文件，现在我们将小强主机远程目录添加到本地遥控端，这样在本地就可以直接图形化操作小强主机上的文件（小强主机相当于本地 UBUNTU 系统的外挂硬盘）。

点击下图位置，添加小强远程目录

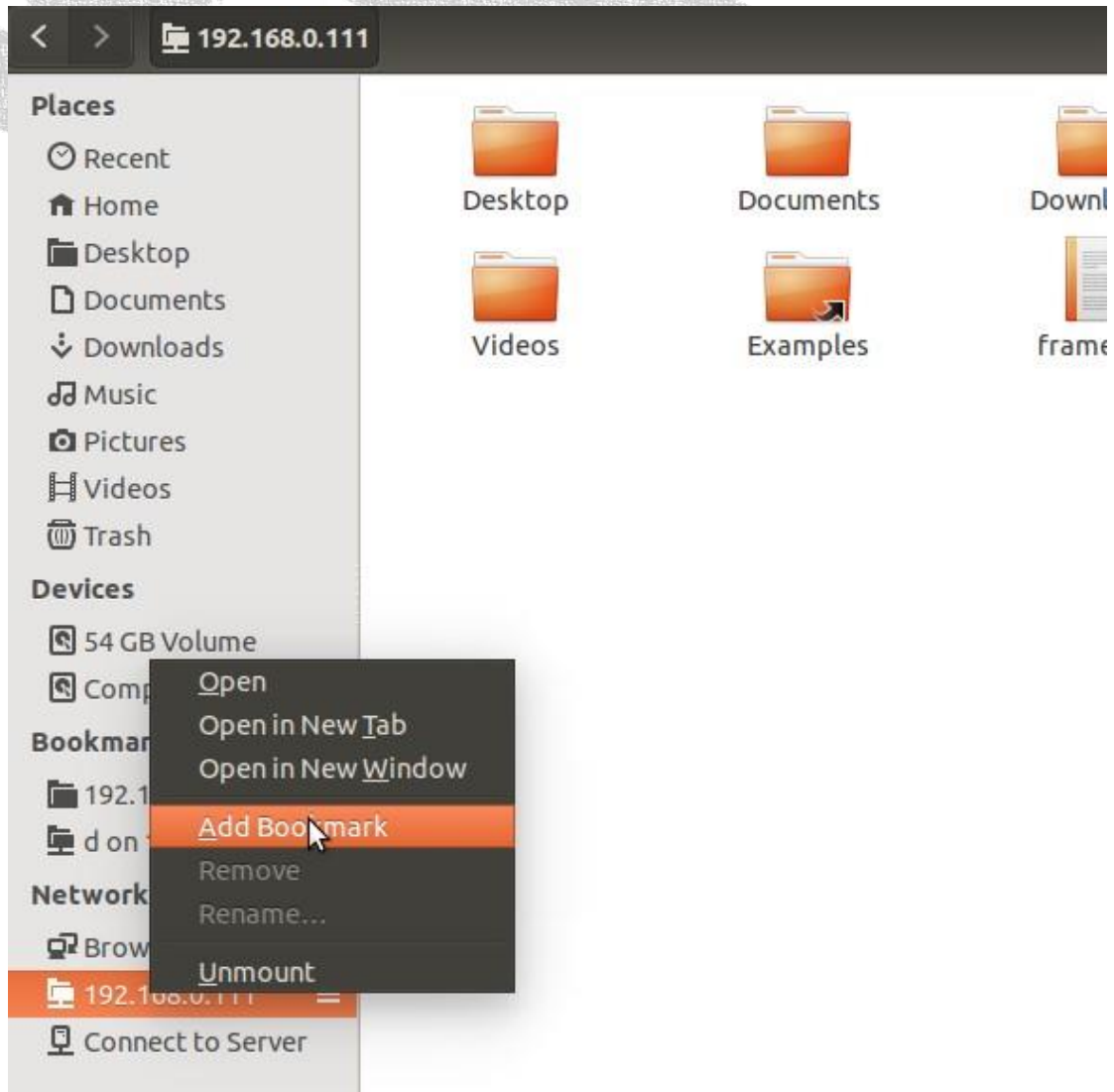


输入小强远程目录，请将 ip 换成上文提到的实际 ip 地址

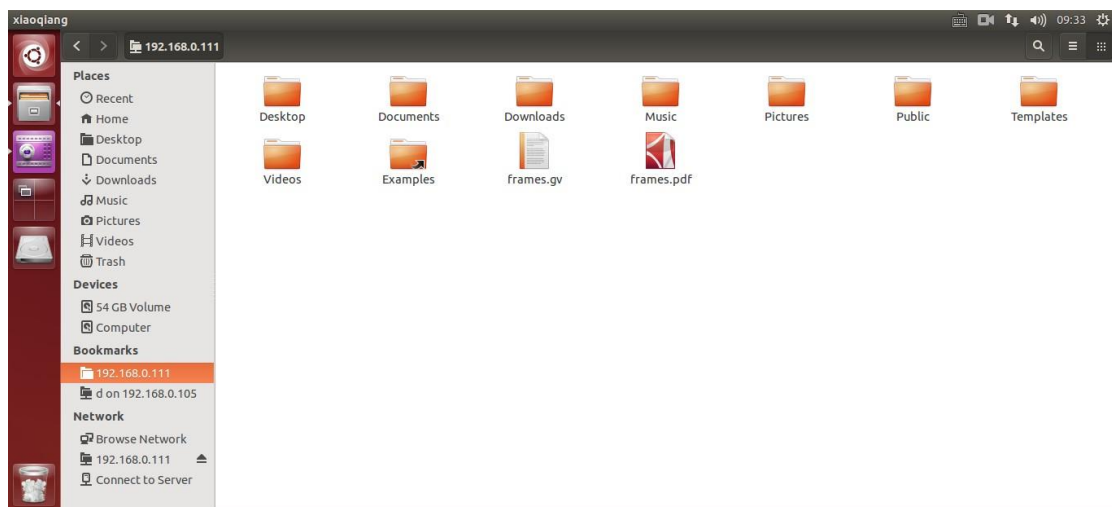


根据提示输入小强主机用户名和密码

一切正常的话，已经打开了小强主机的 home 目录，为了未来使用方便，可以将这个地址添加到 bookmark



下次直接点击这个 bookmark，就能访问小强主机的 home 目录



建议安装 atom 编辑器(小强镜像系统已经安装), 可以方便地对小强主机上的软件包进行代码编辑

```

main.cpp
12 using namespace std;
13
14 int main(int argc, char **argv)
15 {
16     cout<<"welcome to xiaoqiang serial server,please feel free at home!"<<endl;
17
18     ros::init(argc, argv, "xqserial_server");
19     ros::start();
20
21     //获取串口参数
22     std::string port;
23     ros::param::param<std::string>("-port", port, "/dev/ttyUSB0");
24     int baud;
25     ros::param::param<int>("-baud", baud, 115200);
26     cout<<"port:"<<port<<" baud:"<<baud<<endl;
27     //获取小车机械参数
28     double separation=0,radius=0;
29     bool DebugFlag = false;
30     ros::param::param<double>("-wheel separation", separation, 0.37);
31     ros::param::param<double>("-wheel radius", radius, 0.06);
32     ros::param::param<bool>("-debug flag", DebugFlag, false);
33     xqserial_server::StatusPublisher xq_status(separation,radius);
34
35     //获取小车控制参数
36     double max_speed;
37     string cmd_topic;
xqserial_server/src/main.cpp 11
  
```

D. 配置完成, 开始使用

现在您已经具备小强的完整开发使用条件, 例如 SSH 登录小强主机, 开始用键盘遥控小强移动

(1)在本地遥控端打开一个命令终端

(2)通过 ssh 连接小强, 请将 xxx 换成实际的 ip

```
ssh xiaoqiang@xxx.xxx.xxx.xxx
```

(3)启动遥控程序

```
roslaunch nav_test control.py
```

(4)可以开始通过方向键来控制小强的移动了。空格键是停止。Ctrl + C 退出程序。

教程(2) 蓝鲸智能开源软件仓库的使用和 ROS 开机启动任务的配置

小强的所有软件源码都共享在蓝鲸智能的开源仓库里, 任何人任何组织都可以自由下载使用或进行二次开发, [软件仓库](https://github.com/BlueWhaleRobot)地址为: <https://github.com/BlueWhaleRobot>

对于小强用户, 开源仓库中的软件可以直接 `git clone` 到小强的 ROS 工作目录里, 然后就可以直接用 ROS 的工具链 `catkin_make` 编译使用。小强的 ROS 工作目录为: `/home/xiaoqiang/Documents/ros/src`

下文将以开源仓库中的 `startup` 软件包为例, 演示开源仓库的完整使用过程。

2.1 STARTUP 软件包功能介绍

小强主机开机后, 会自动启动名字为 `startup` 的 linux 服务脚本, `startup` 服务脚本运行时去启动 `startup` 软件包中的 `startup.launch` 文件 在 `ubuntu` 系统中注册的复制品。因此我们通过修改 `startup` 软件包中的 `startup.launch` 文件, 然后将这个文件在 `ubuntu` 系统中注册为 `startup` 服务, 就能控制小强主机的开机启动任务了。

2.2 在小强主机中下载安装 STARTUP 软件包

a. 在本地遥控端 `ssh` 连接小强主机, 参考上篇教程的配置

```
ssh xiaoqiang@192.168.x.x
```

b. 进入小强 ROS 工作目录, 查看是否有 `startup` 文件夹

```
cd Documents/ros/src/  
ls
```

如果存在, 说明已安装好 `startup` 软件包, 可以直接进行下面的操作三
如果想和开源仓库同步更新这个 `startup` 软件包, 请输入如下命令

```
cd startup  
git stash  
git pull  
cd ..
```

2.3 修改软件包中 LAUNCH 文件夹内的 STARTUP.LAUNCH 文件

利用上篇教程安装的 `atom` 编辑器, 在本地遥控端直接编辑这个文件 (需要远程访问小强的主机文件目录, 请参考上篇基础操作教程进行配置)

在上图箭头区域，添加或删除你需要启动的 ROS launch 文件及 ROS node，这些项目在下文将被添加进小强主机的开机启动项里，小强下次开机自动运行这些项目。最后保存退出

2.4 将 STARTUP.LAUNCH 文件在小强主机中注册为 STARTUP 开机启动服务

接着二中的 ssh 窗口输入

a. 首先将之前注册的 startup 服务停止和删除

```
sudo service startup stop
rosrun robot_upstart uninstall startup
```

b. 重新注册 startup 开机启动服务

```
rosrun robot_upstart install startup/launch/startup.launch
```

2.5 远程重启小强主机，查看开机启动项是否正常加载

接着上文的 ssh 窗口输入

a. 下发重启命令

```
sudo shutdown -r now
```

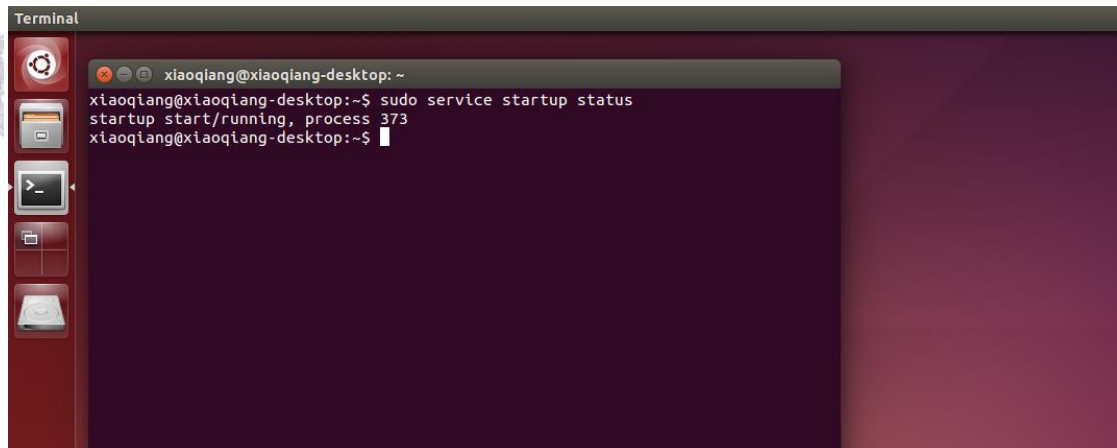
b. 重新 ssh 连接

```
ssh xiaoqiang@192.168.x.x
```

c. 查看 startup 服务状态

```
sudo service startup status
```

正常的话会显示 startup start/running 如下图所示

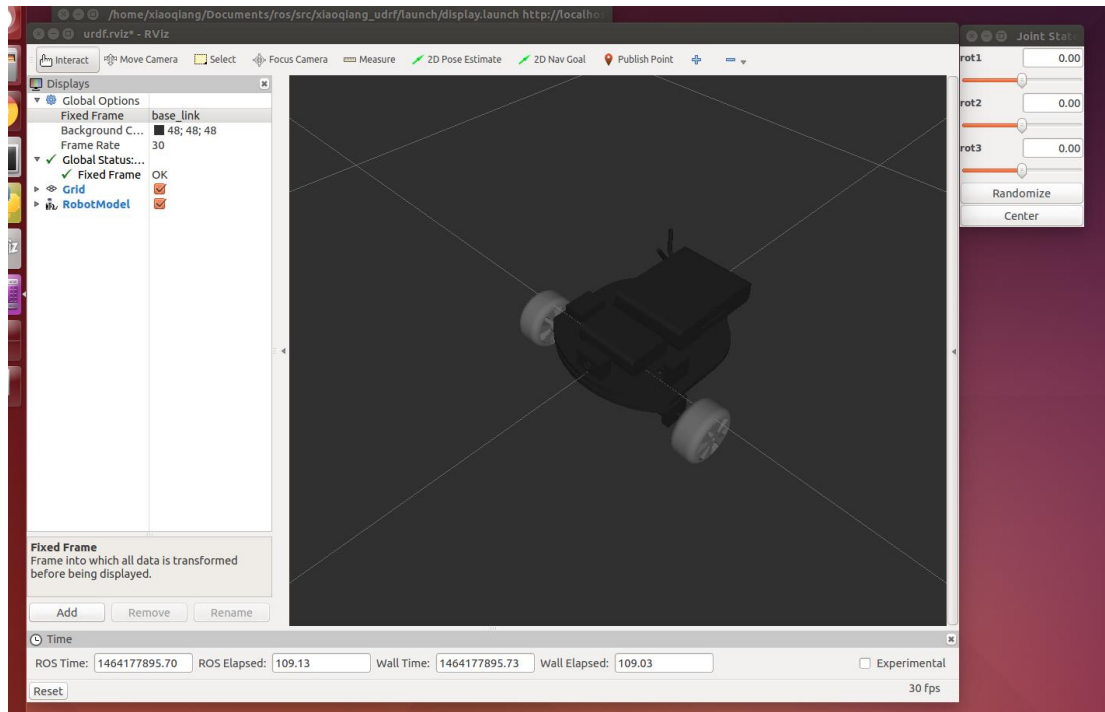


d.还可以进一步查看相关的 topic 是否已经发布出来

```
rostopic list
```


教程(3) 在 rviz 中显示小强机器人模型

实时显示机器人当前姿态是一件很酷的事情，在 ROS 中借助 rviz 可以轻松实现这个目标。先看最后的效果：完整视频请[下载](#)本文件.mp4



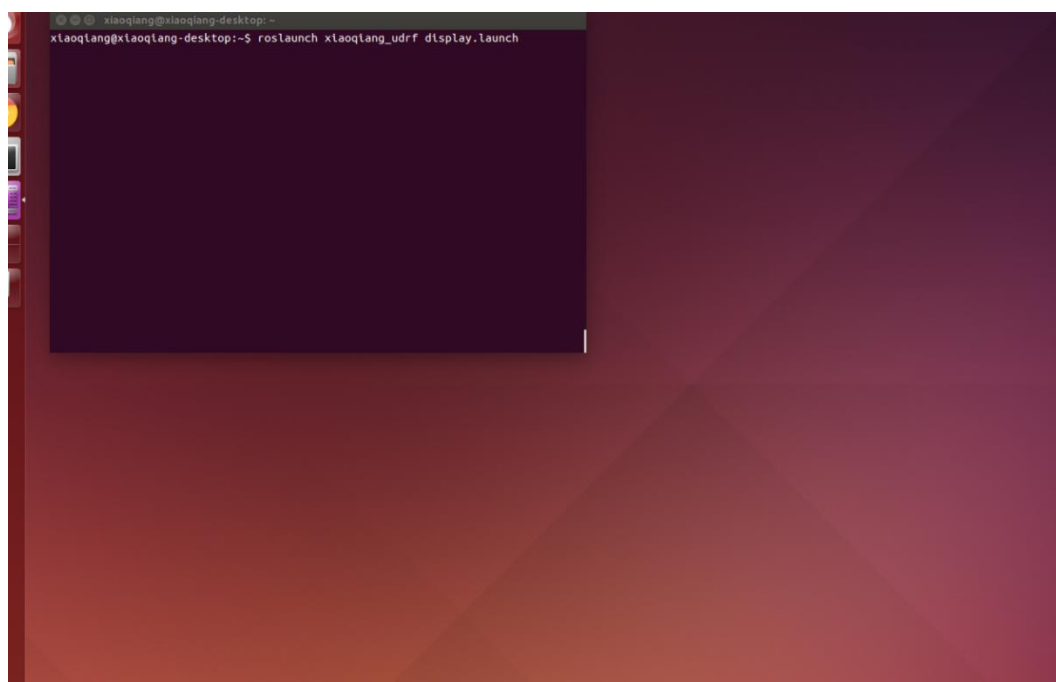
将小强主机接入显示器和键盘，开机后，打开终端，先关闭开机任务

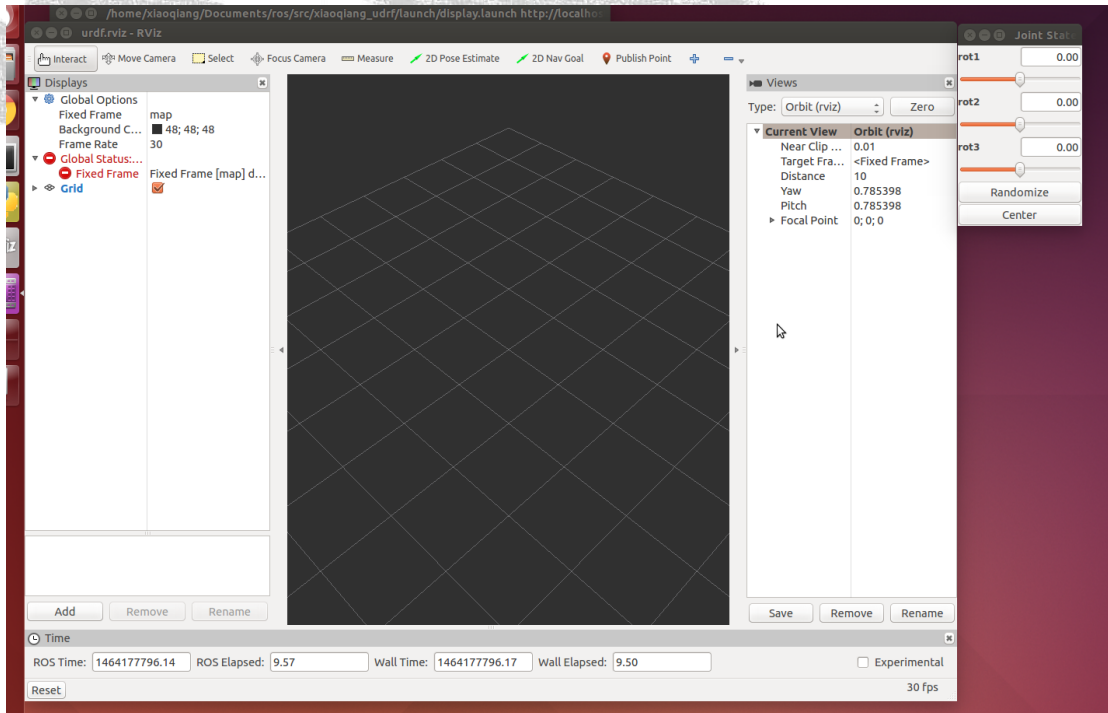
```
sudo service startup stop
```

```
roscore
```

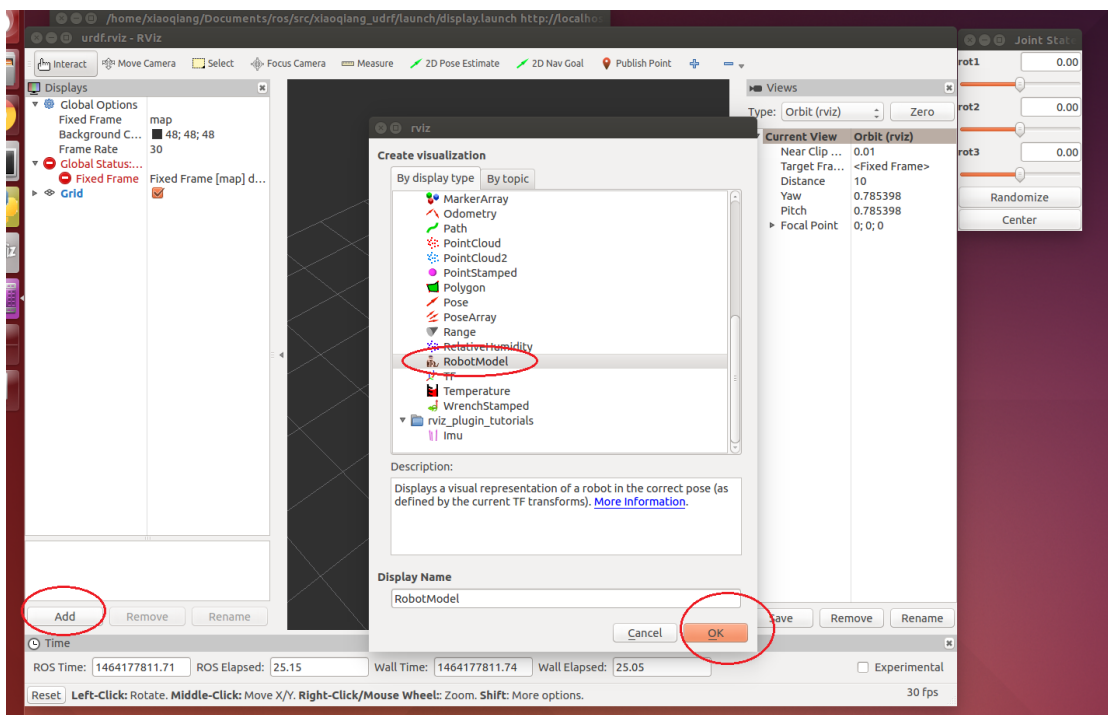
在小强主机上新开一个终端，启动这个软件包

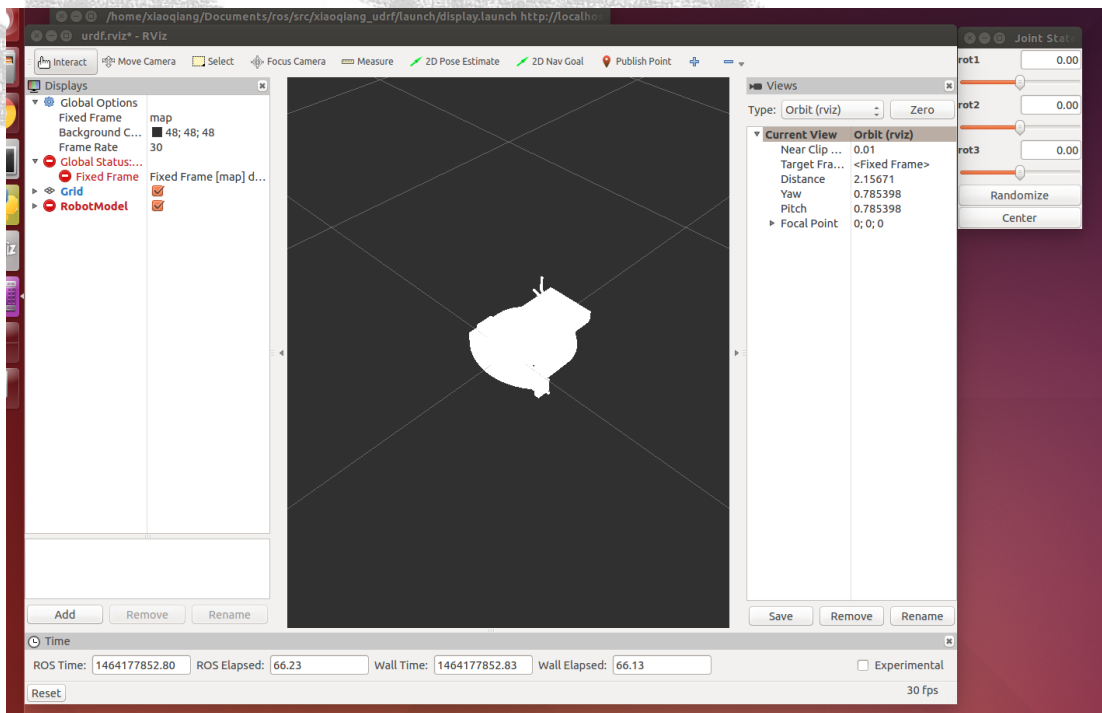
```
roslaunch xiaoqiang_udrf display.launch
```



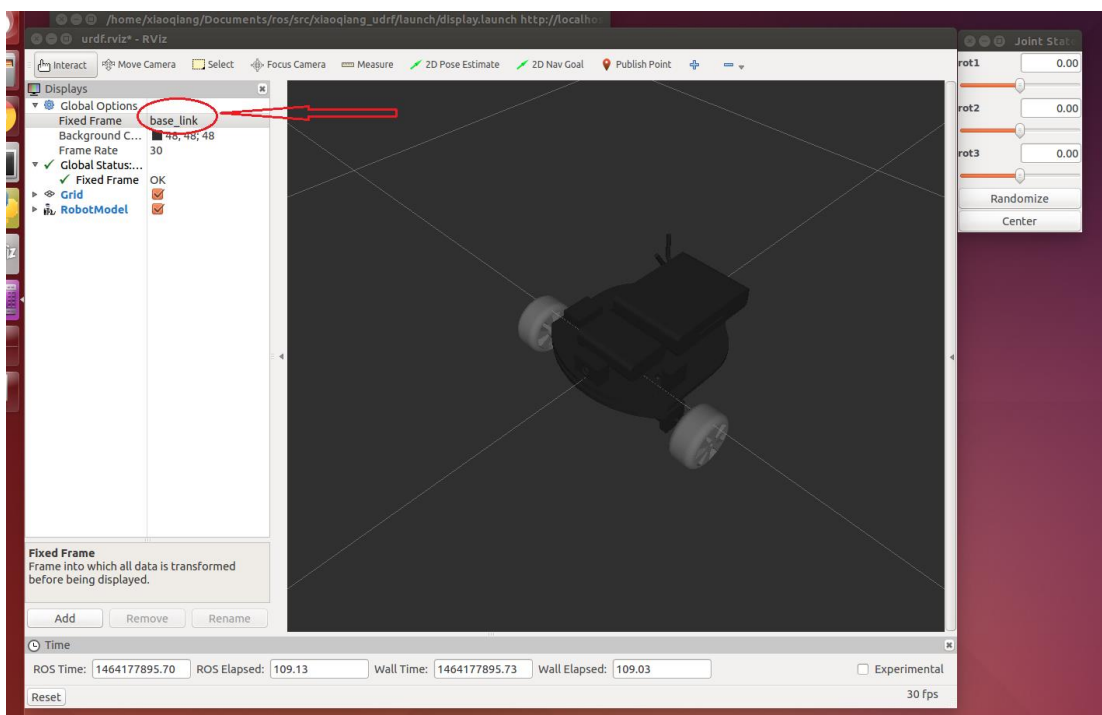


此时发现没有任何显示，需要添加 rviz 显示项目





还是有问题，整个模型透明发白，这是因为 rivz 中的全局坐标系“fixed frame”设置的不合适，将 map 改成 base_link 后即可正常显示

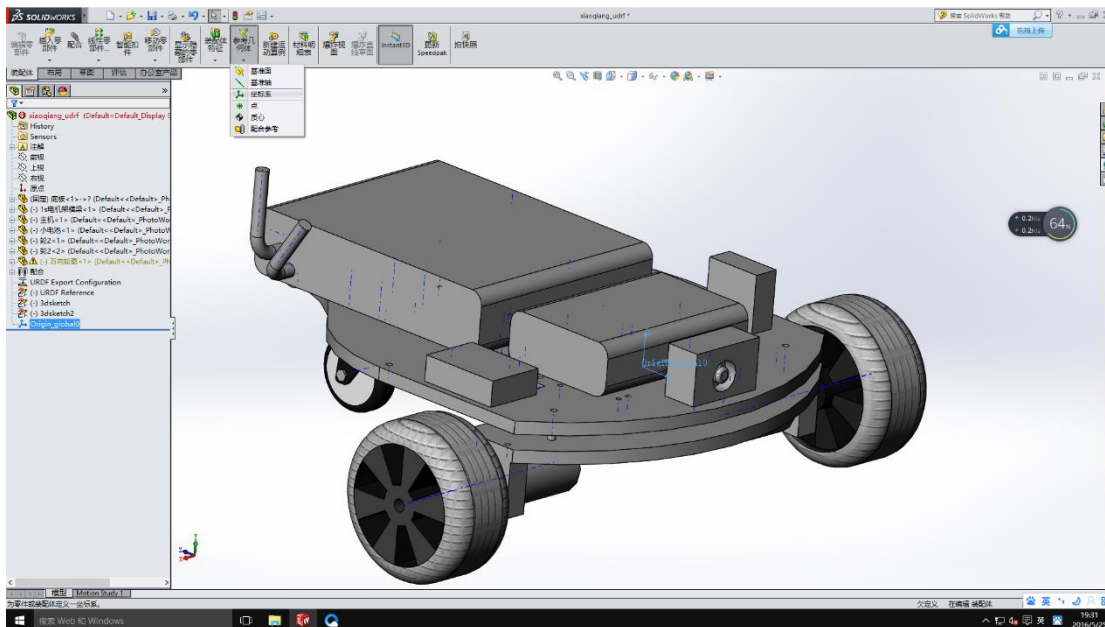


现在操作右上角的滑动条就可以使相应的轮子转动。

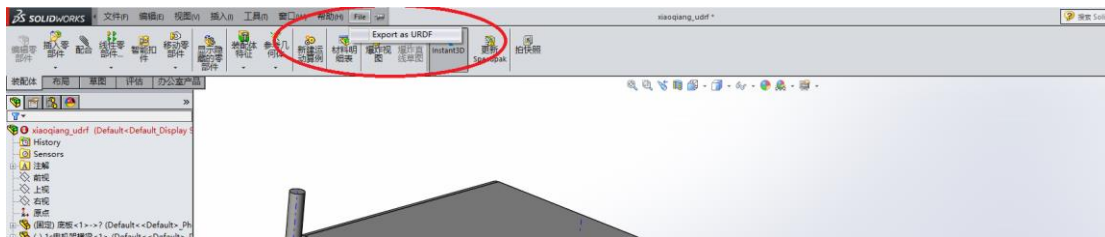
上面我们简单演示了 **rviz** 显示 **urdf** 模型的使用方法，下文将详细介绍在 **windows** 系统下用 **solidworks** 制作 **urdf** 模型包的整个过程。

用 **solidworks** 建立小车模型，并下载安装好 **solidwork** 转 **urdf** 插件(下载地址：http://wiki.ros.org/sw_urdf_exporter)。

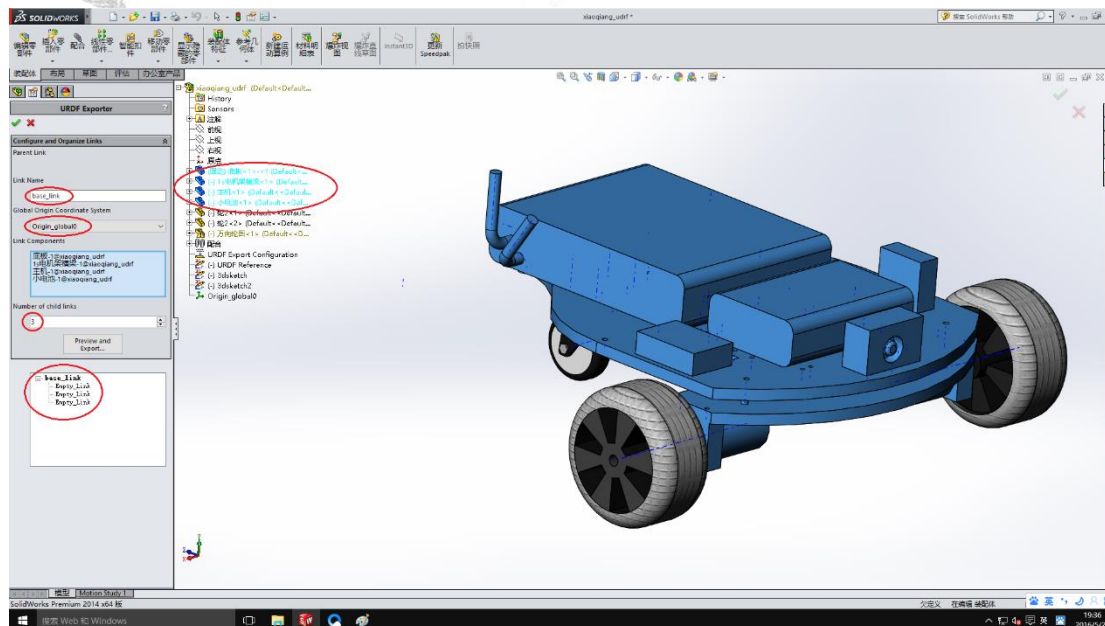
制作好模型后，需要补加一个坐标系，这个坐标系在下文将作为整个 **urdf** 模型的基准坐标系（即 ROS 中的 **base_link frame**）。



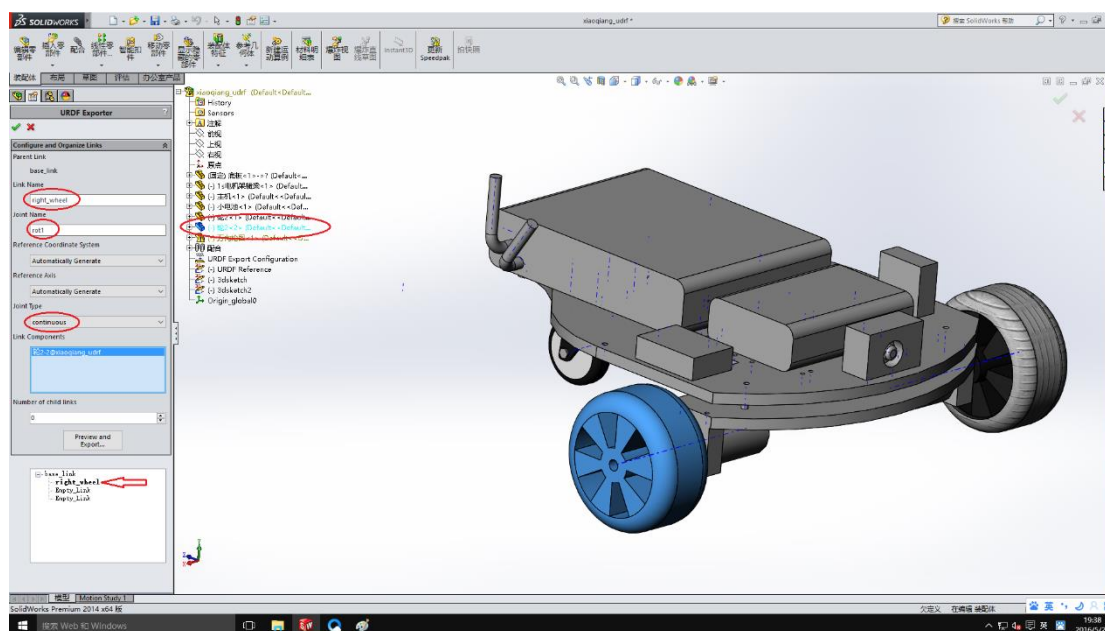
打开 **urdf** 插件



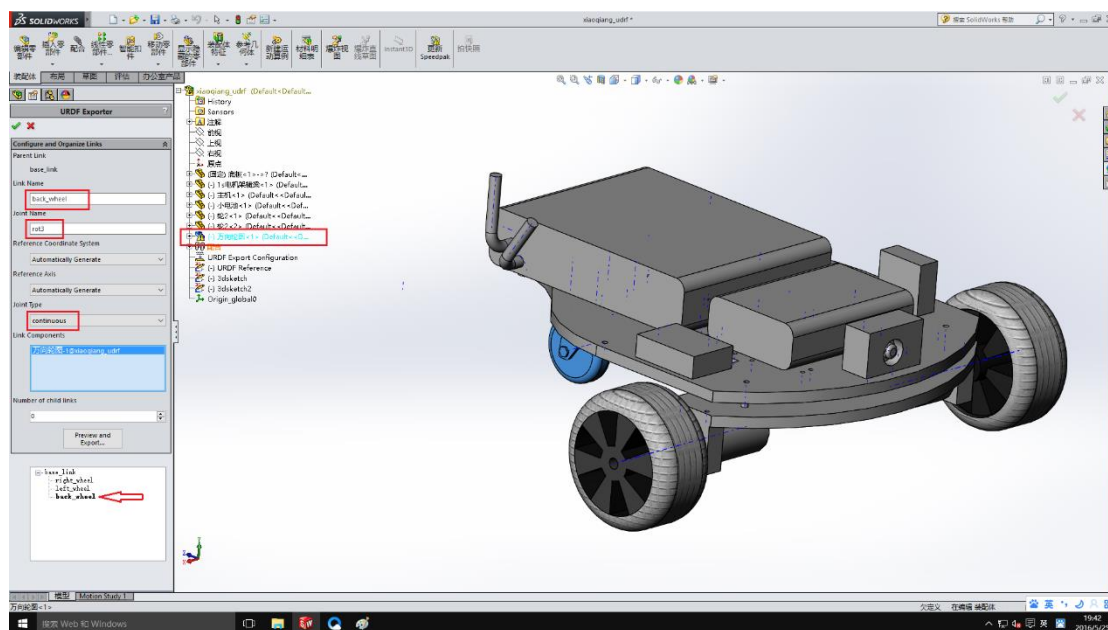
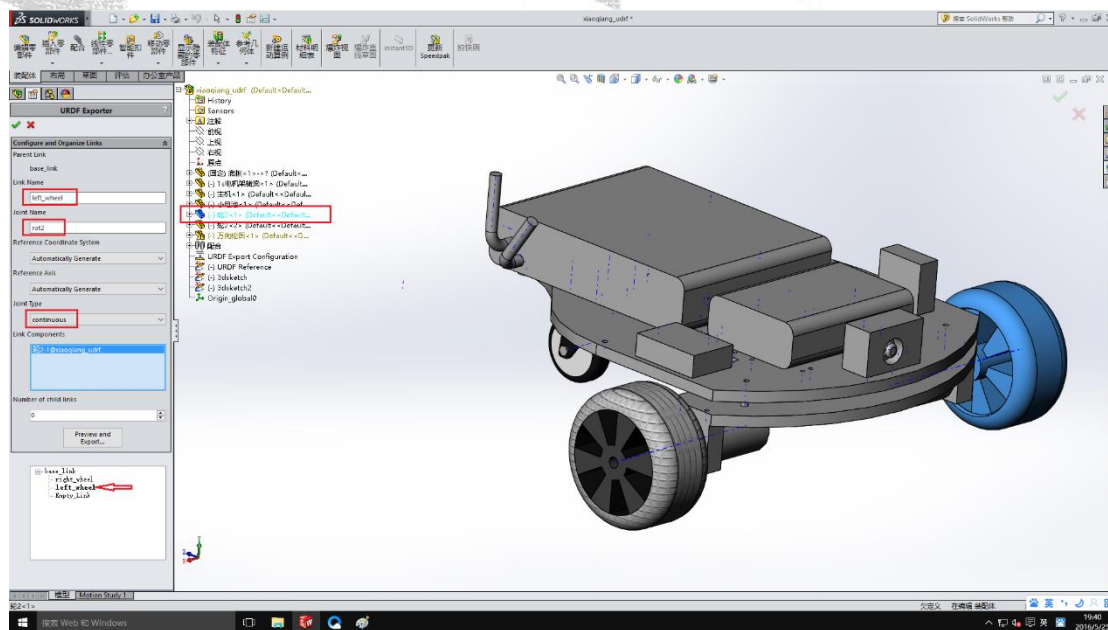
小强有两个驱动轮和一个从动轮，所以整个模型需要 3 个 link, 3 个 joint. 首先编辑 base_link, 留意上文中全局坐标系，图片中红色区域就是需要自己点击或者修改的项目



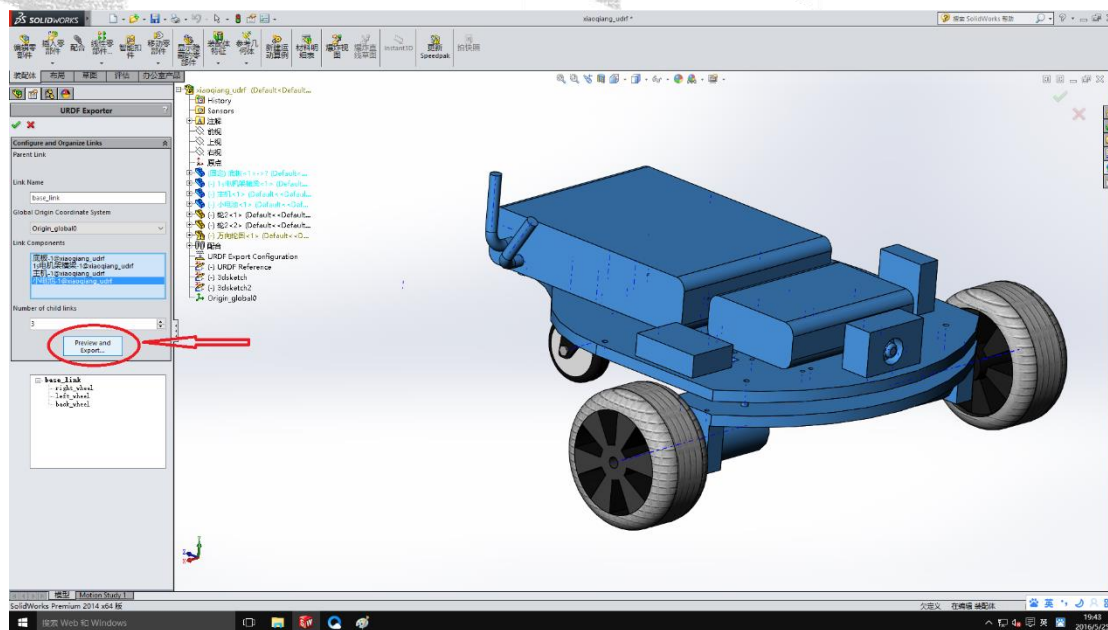
然后是右轮



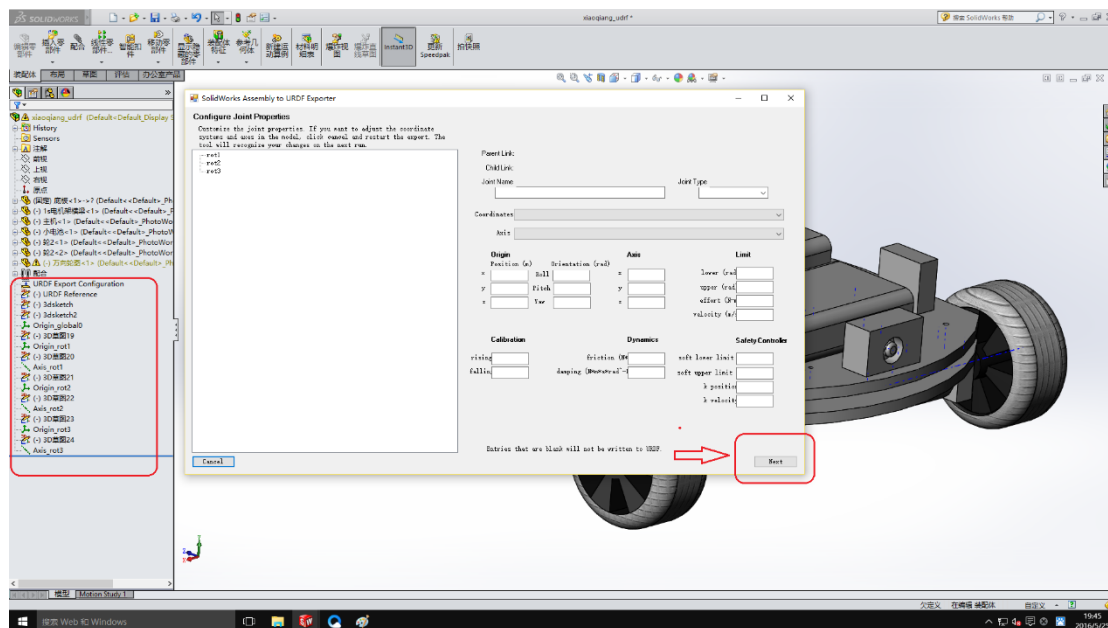
继续左轮和后轮

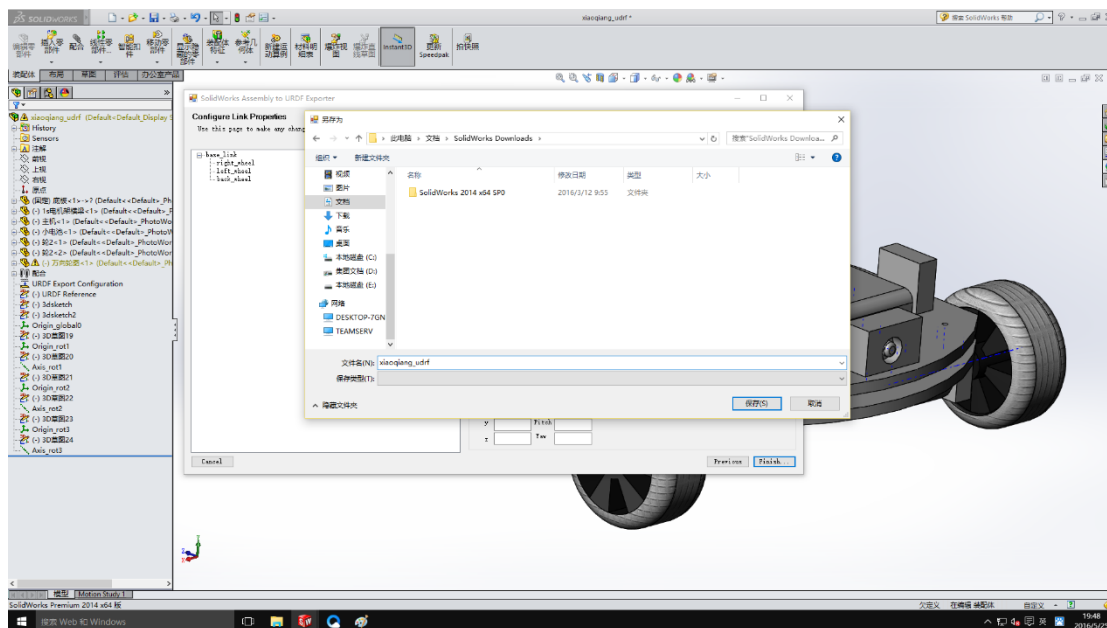
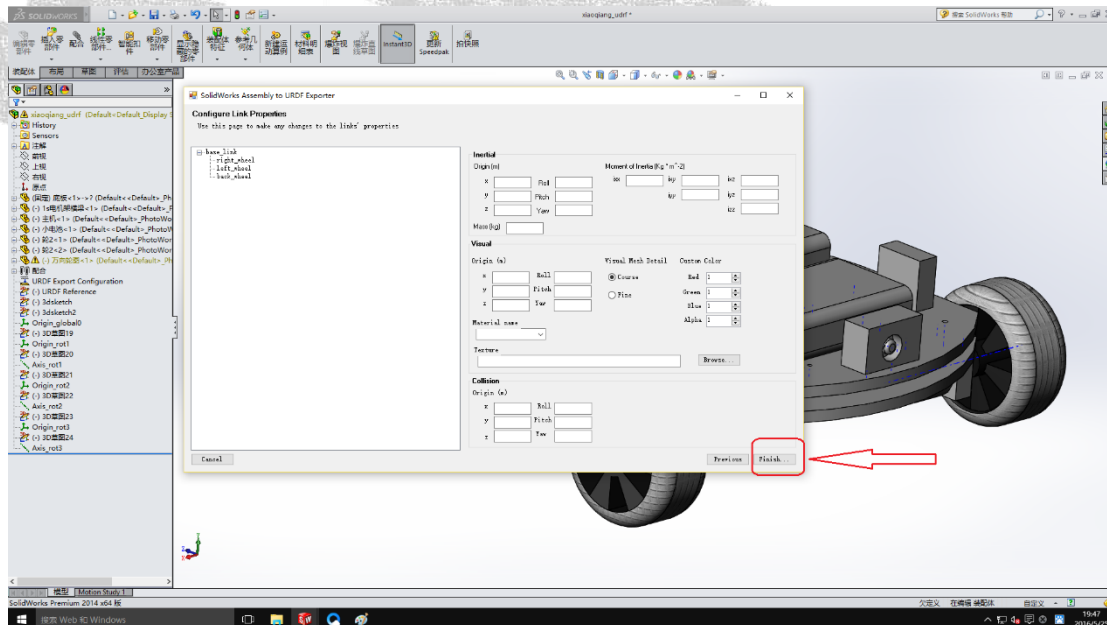


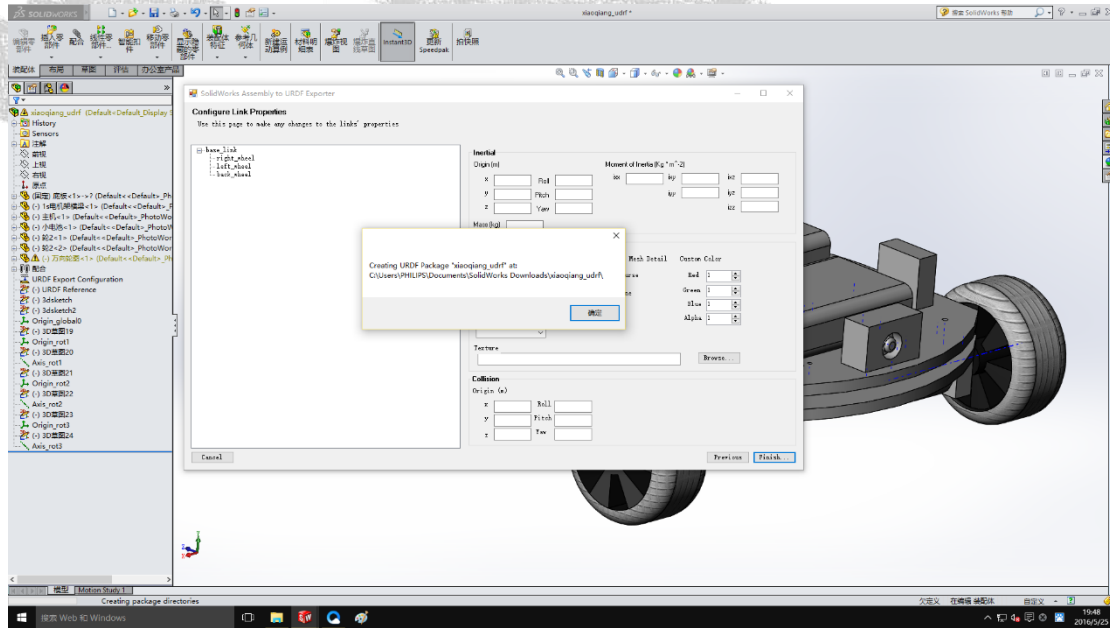
现在全部设置完成，开始导出



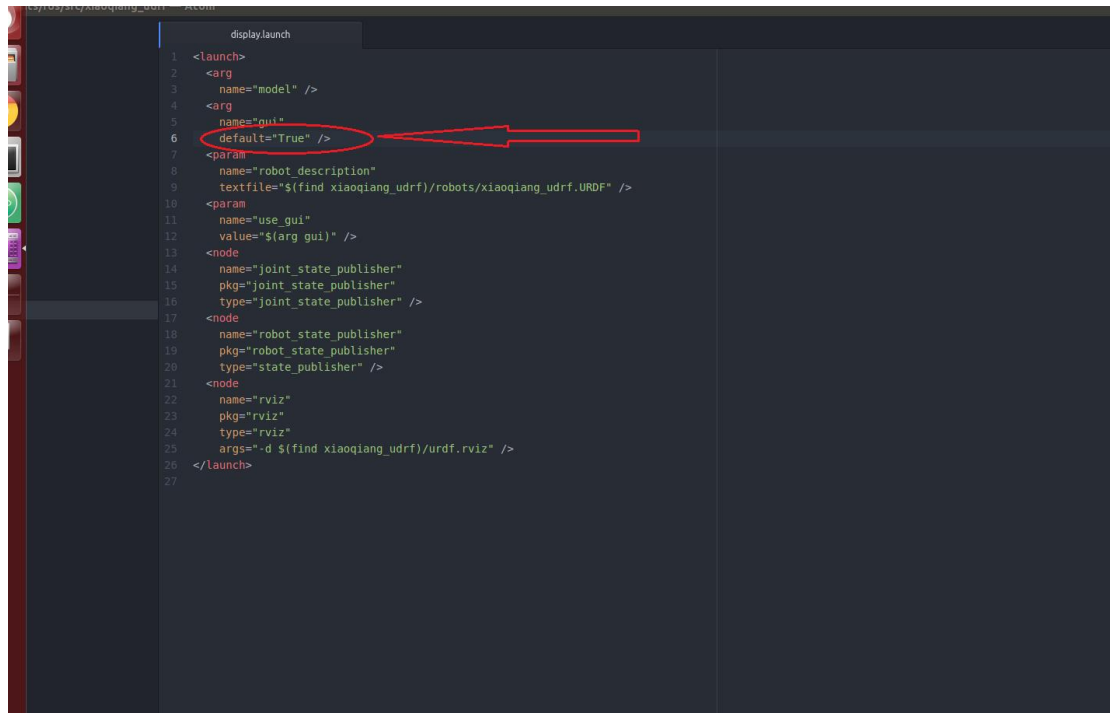
一路 next 和确定下去







现在我们已经获到了小强的 urdf 文件，生成的整个文件夹是一个 ROS 包， 修改 launch 文件夹内的 display.launch 文件 ,false 改 true

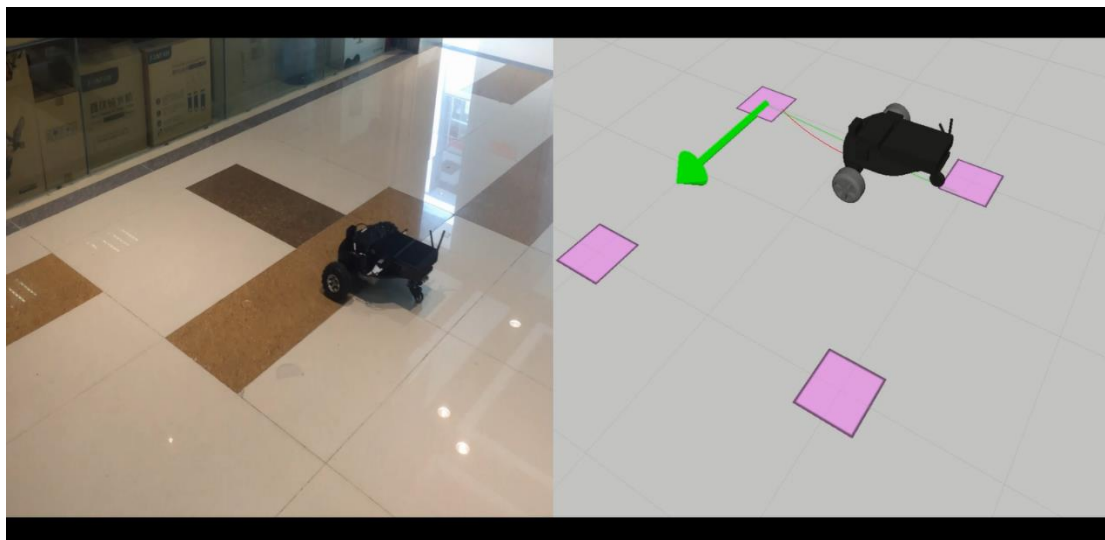


将这个 ros 包复制到 ROS 工作空间中，catkin_make 编译后就可以用本文开头的方法测试使用了。

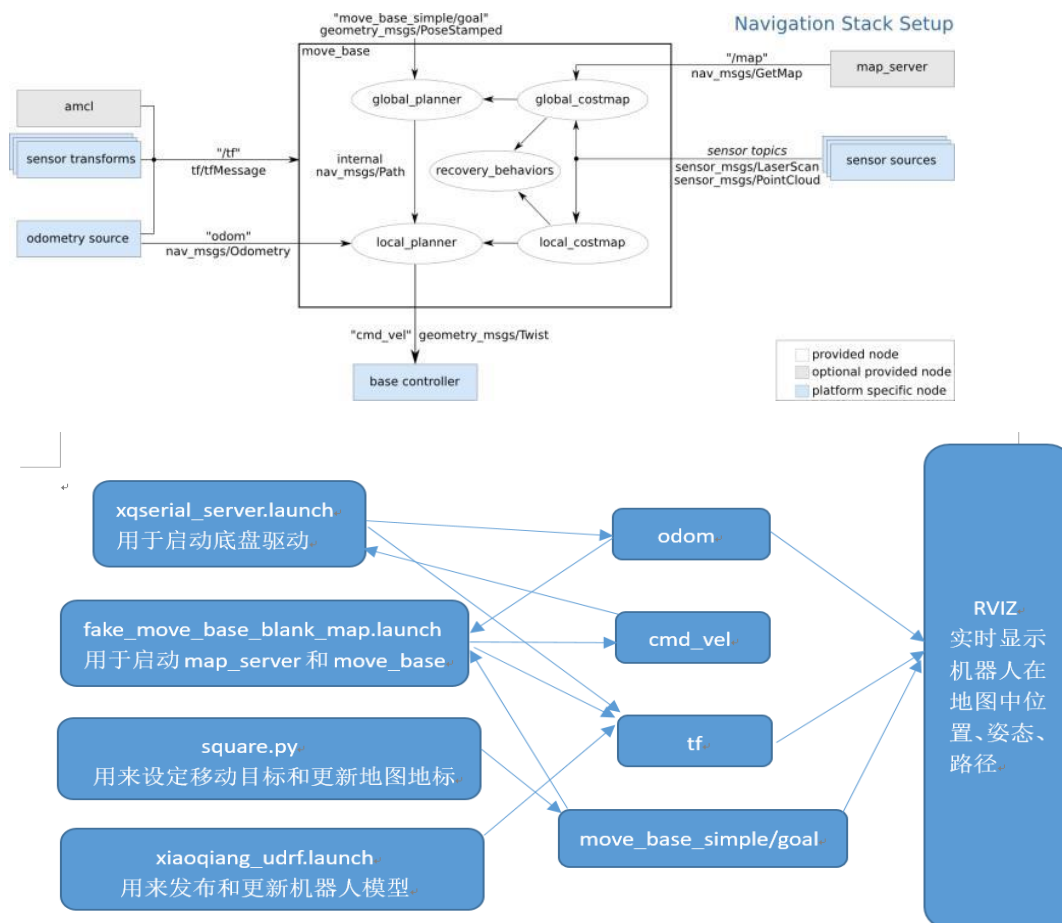
教程(4) 惯性导航自主移动测试

上一篇文章中，我们已经制作出了小强的 3d 模型，接下来开始测试小强的惯性导航功能。这里的惯性导航，是指利用小强自身佩戴的惯性传感器（加速度和陀螺仪）和底盘编码器信息进行定位和移动。需要的 ROS 软件包有：1.底层驱动 `xqserial_server`, 2.机器人模型包 `xiaoqiang_udrf`, 3.惯性导航测试软件包 `nav_test`。

先上教程最后效果图，完整视频在这里 [selfmove_rviz1.mp4](#)



整个导航测试的内部框架如下两图所示：如果对 ROS 导航框架不熟悉，请移步这篇教程



1. ssh 方式在小强主机上完成的操作

请确保小强已经正常启动，小强主机正常启动完成后会自动运行上面提到的三个软件包，不需要手工启动相应的 launch 文件。

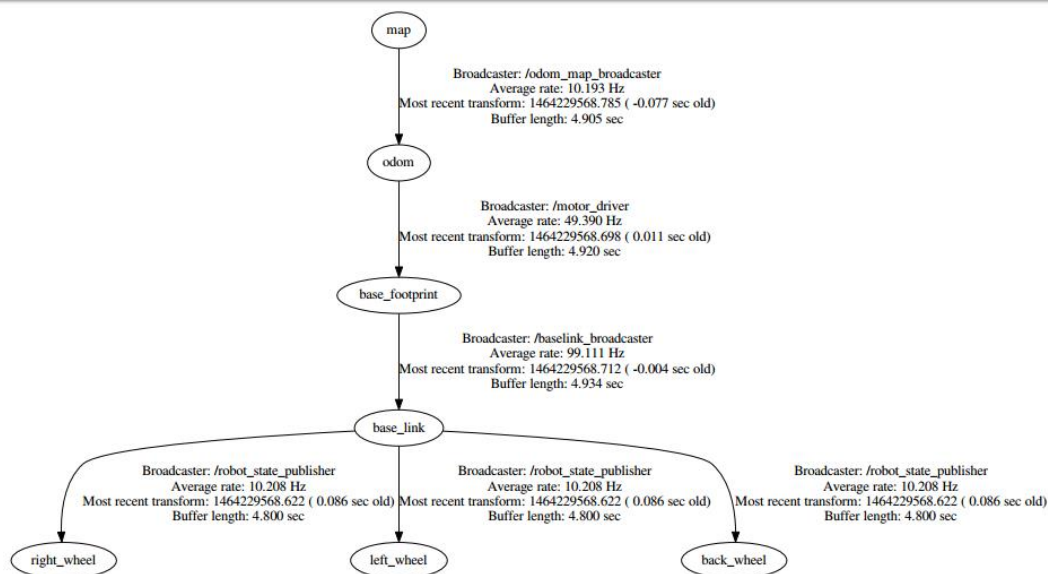
a. 新开一个终端，启动导航基础程序

```
ssh -X xiaoqiang@192.168.xxx.xxx
roslaunch nav_test fake_move_base_blank_map.launch
```

b. 新开一个终端，检查是否所有 tf 都已经就位

```
ssh -X xiaoqiang@192.168.xxx.xxx
roslaunch tf view_frames
evince frames.pdf
```

正常会显示下图



2. 在本地遥控端上完成的操作

本地遥控端必须是安装好 ROS jade 版本的 ubuntu 系统, [请参考教程(1)中的 1.2 节安装小强系统镜像], 同时保证和小强主机在同一个局域网内。因为需要在本地窗口用 rviz 显示小强姿态和路径轨迹 (ssh 中不能直接打开 rviz), 所以需要使用 ros 的分布式网络配置方案。同时在本地遥控端也需要安装好机器人模型包 xiaoqiang_udrf。

概括来说: 本地遥控端打开自己的 rviz, 接收显示小强主机上的 topic, 而小强模型数据则直接从本地获取。具体过程如下:

a. 在本地开一个命令行终端, 在本地的 hosts 文件内添加小强的 ip

```
sudo gedit /etc/hosts
```

添加

```
xxx.xxx.xxx.xxx xiaoqiang-desktop #请将 xx 改成小强的实际 ip 地址
```

b. 新开一个命令终端输入

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
```

继续执行

rostopic list

如果可以看到小强的 topic 了,就说明配置成功。

c. 安装了小强镜像的用户请不要执行本步骤, 安装模型软件包, 更新本地 ROS 包环境变量, 因为需要从本地读取模型数据

```
mkdir ~/Documents/ros/src
```

```
cd ~/Documents/ros/src
```

```
catkin_init_workspace
```

```
git clone https://github.com/BlueWhaleRobot/xiaoqiang\_udrf.git
```

```
#小强 pro 用户切换到 master 分支
```

```
cd xiaoqiang_udrf
```

```
git checkout master
```

```
#小强 mini 用户切换到 mini 分支
```

```
cd xiaoqiang_udrf
```

```
git checkout mini
```

```
#编译完成安装
```

```
cd ..
```

```
cd ..
```

```
catkin_make
```

d. 打开 rviz 图形界面

```
source ~/Documents/ros/devel/setup.sh
```

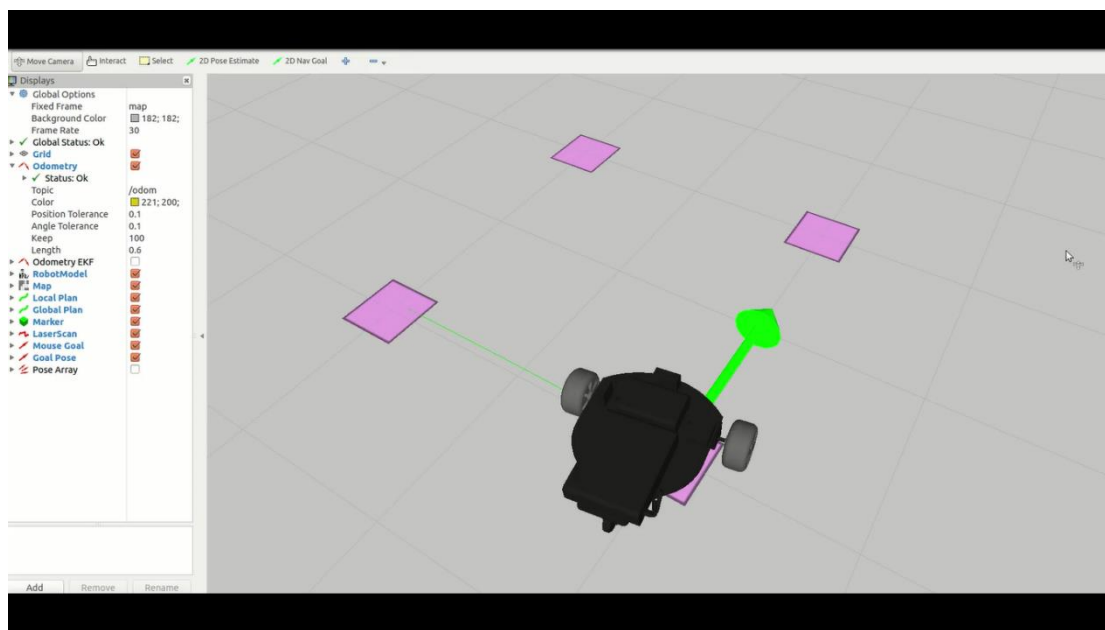
```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
```

```
rviz
```

当窗口打开后, 点击左上角的 file->open, 选择小强里的

/home/xiaoqiang/Documents/ros/src/nav_test/config/nav.rviz 文件。

这时界面应该如下图显示,关于如何访问小强主机上的文件, 请参考之前的教程。



3. 在小强主机远程 ssh 窗口内完成最后操作

roslaunch nav_test square.py

4.现在在 rviz 中就能看到文章开头的视频效果了

教程(5)___小强遥控图传 app 安卓版

2017 年 3 月份之前收到小强的用户，安装前请参考帖子[\[升级软件包以支持小强图传遥控 app\]](#)

版本 1 有运动遥控功能，小屏图传显示
[xiaoqiang-with-control.apk](#)



版本 2 没有遥控功能，全屏图传显示
[xiaoqiang-no-control.apk](http://www.bwbot.org/xiaoqiang-no-control.apk)



使用方法:

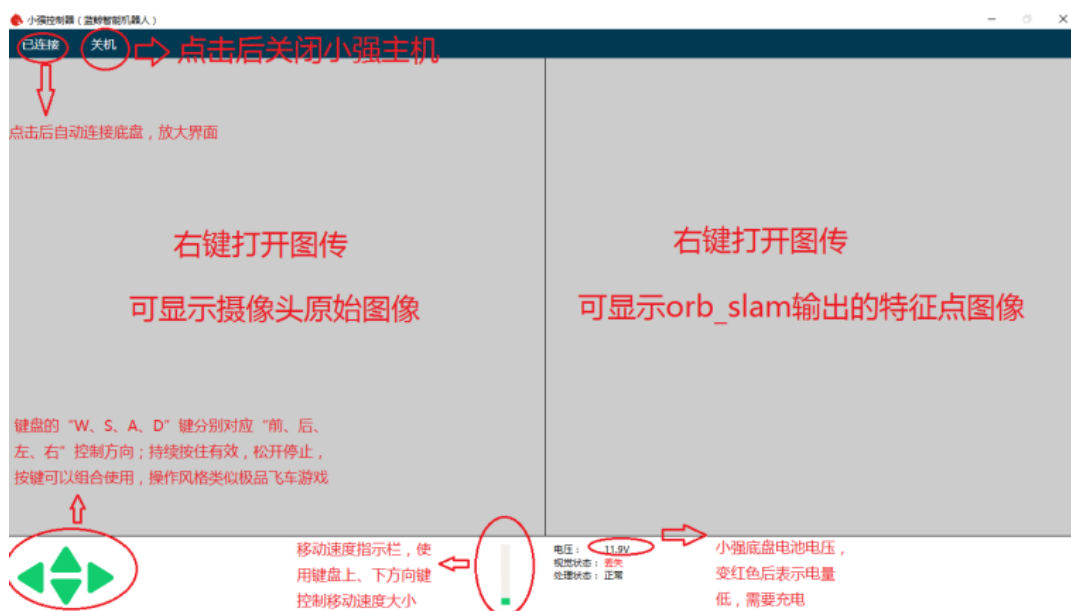
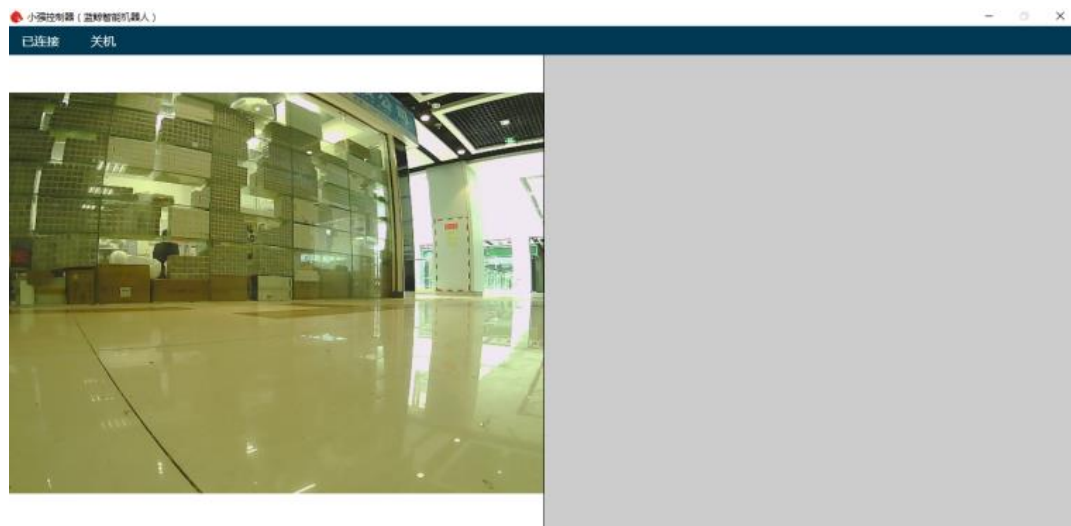
1. 保证小强和遥控用手机在同一局域网内
2. 在小强上开启服务端程序
3. 打开 app, 如果一切正常就可以看到小强的电压显示和小强的图像数据了。如果没有数据可以尝试点击重连接按钮。

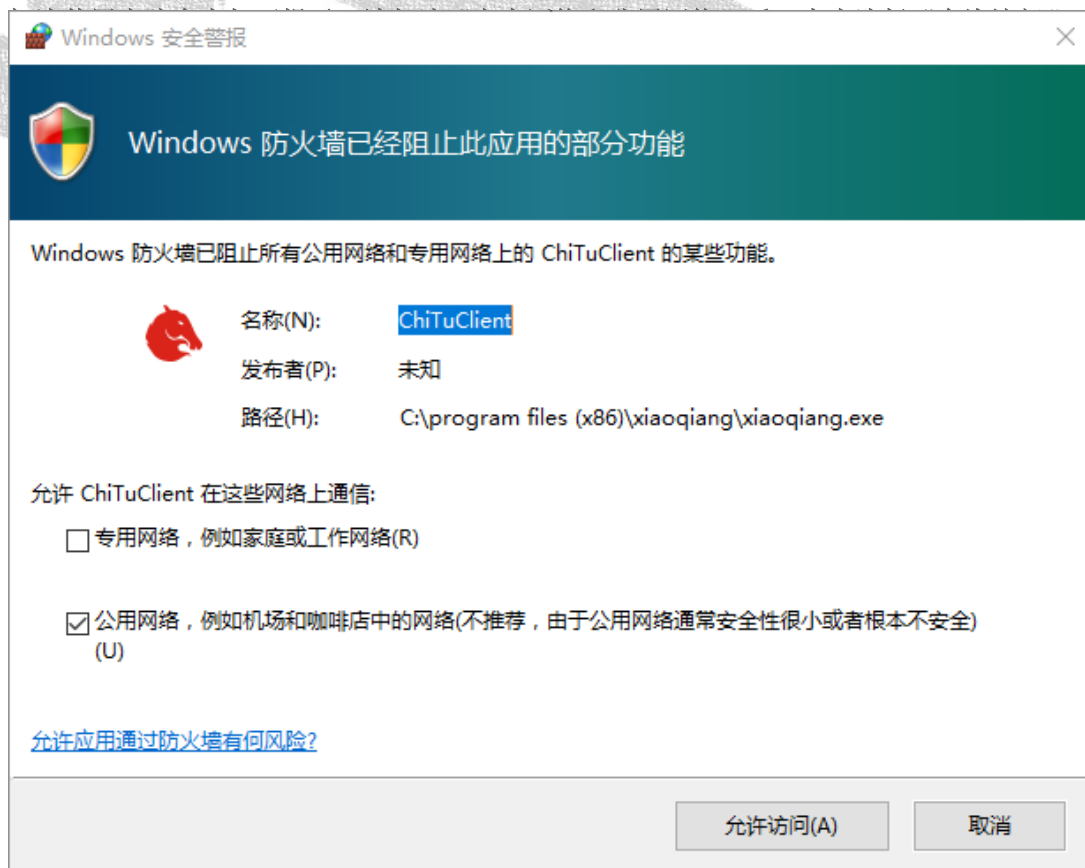
教程(6) 小强遥控图传 windows 客户端

2017 年 3 月份之前收到小强的用户，安装前请参考帖子[\[升级软件包以支持小强图传遥控 app\]](#)

软件安装包下载连接

下载解压后根据“安装说明.txt”提示完成安装，软件界面操作如下图所示





教程(7)___使用 ps3 手柄控制小强

原理：本教程涉及 3 个包，`ps3joy` 负责将 ps 3 蓝牙接受信号转换成标准的 linux 设备(/dev/input/js0)，`joy_node` 节点负责将上述 joy 设备数据转换成 ros 中的 joy 数据类型，`turtlebot_teleop_joy` 负责将上述 joy 数据 topic 转换成小车运动指令 /cmd_vel

操作步骤：

0. 第一次使用手柄，需要将手柄与蓝牙接受器进行绑定，以后可以直接从步骤 1 开始

绑定方法参考[\[原装和国产 ps3 手柄 ros 驱动程序\]](#)中的“快速使用方法步骤 1”

1. 启动 PS3JOY，将 PS3 手柄与蓝牙接受器配比

```
#确保蓝牙接收器已经插入主机usb口
```

```
sudo bash
```

```
roslaunch ps3joy ps3joyfake_node.py
```

正常会出现类似下文的配对提示

```
root@xiaoqiang-desktop:~# roslaunch ps3joy ps3joyfake_node.py
```

```
No inactivity timeout was set. (Run with --help for details.)
```

```
Waiting for connection. Disconnect your PS3 joystick from USB and press the pairing button.
```

按下下图中的手柄配对键



配对成功的话，手柄会震动一下，同时上面的窗口会输出类似下面的结果

```
root@xiaoqiang-desktop:~# roslaunch ps3joy ps3joyfake_node.py
```

```
No inactivity timeout was set. (Run with --help for details.)
```

```
Waiting for connection. Disconnect your PS3 joystick from USB and press the pairing button.
```

```
Connection activated
```

2. 启动 JOY_NODE 和 TURTLEBOT_TELEOP_JOY

```
roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
```

正常启动后如下图所示

```

/home/xiaoqiang/Documents/ros/ros/turtlebot/turtlebot_teleop/launch/ps3fakexiaoqiang_teleop.launch http://localhost:11311
xiaoqiang@xiaoqiang-desktop:~$ roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
WARNING: package name "billinearplanner" does not follow the naming conventions. It should start with a lower case letter and only contain lower case letters, digits and underscores.
... logging to /home/xiaoqiang/.ros/log/c0991d3e-b139-11e6-8571-3b773a3789a8/roslaunch-xiaoqiang-desktop-6737.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt.
Done checking log file disk usage. Usage is 43GB.
started roslaunch server http://xiaoqiang-desktop:37371/
SUMMARY
-----
PARAMETERS
-----
 * /roscppcore: jode
 * /rosversion: 1.11.20
 * /turtlebot_teleop_joystick/axis_angular: 0
 * /turtlebot_teleop_joystick/axis_deadman: 10
 * /turtlebot_teleop_joystick/axis_linear: 1
 * /turtlebot_teleop_joystick/scale_angular: 0.4
 * /turtlebot_teleop_joystick/scale_linear: 0.4
NODES
----
 /
  joystick (joy/joy_node)
  turtlebot_teleop_joystick (turtlebot_teleop/turtlebot_teleop_joy)
ROS_MASTER_URI=http://localhost:11311
core service [/roscpp] found
WARNING: package name "billinearplanner" does not follow the naming conventions. It should start with a lower case letter and only contain lower case letters, digits and underscores.
process[turtlebot_teleop_joystick-1]: started with pid [6755]
process[joystick-2]: started with pid [6756]

```

ps3fakexiaoqiang_teleop.launch 文件内容如下

```
<launch>
```

```
<node pkg="turtlebot_teleop" type="turtlebot_teleop_joy" name="turtlebot_teleop_joystick">
```

```
<param name="scale_angular" value="0.4"/>
```

```
<param name="scale_linear" value="0.4"/>
```

```
<param name="axis_deadman" value="10"/>
```

```
<param name="axis_linear" value="1"/>
```

```
<param name="axis_angular" value="0"/>
```

```
<param name="axis_enbar" value="12"/>
```

```
<param name="axis_disenbar" value="14"/>
```

```
<remap from="turtlebot_teleop_joystick/cmd_vel" to="/cmd_vel"/>
```

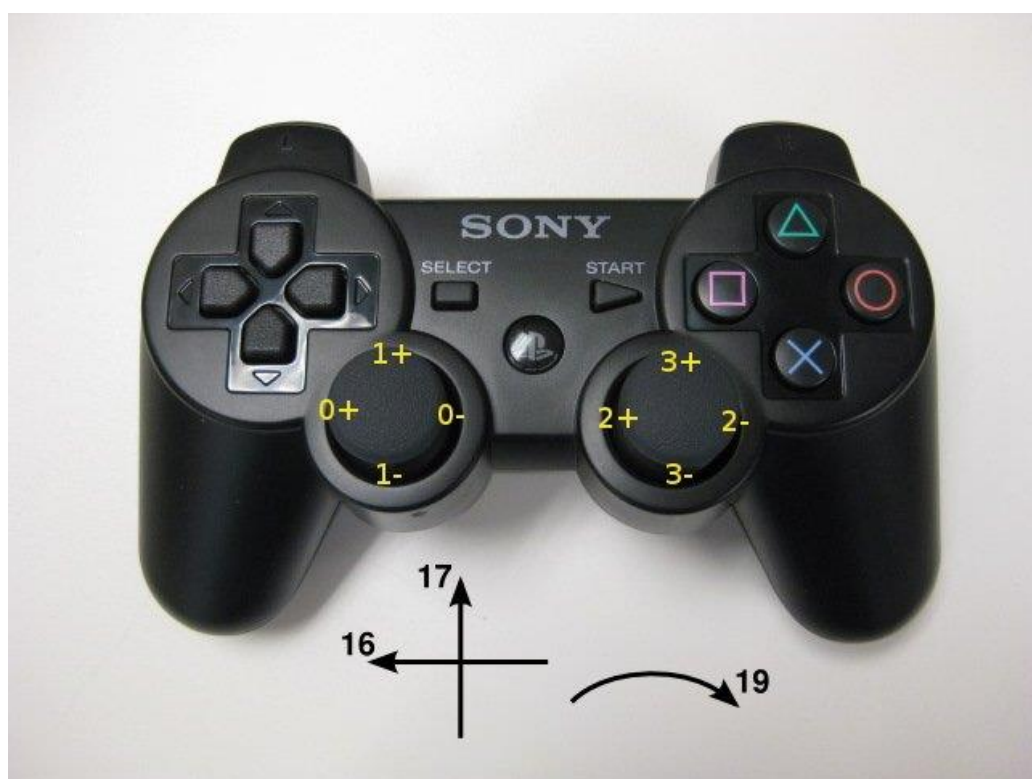
```
<remap from="turtlebot_teleop_joystick/joy" to="/joy"/>
```

```
</node>
```

```
</launch>
```

上述 launch 文件中的参数分别对应直线速度最大值(scale_linear)和角速度最大值(scale_angular), 油门离合键(axis_deadman)、前进后退轴(axis_linear)、左右转轴(axis_angular), 底盘红外使能键(axis_enbar), 底盘红外关闭键(axis_disenbar), 这些控制按键、摇杆的映射关系。

3. 保持按住手柄油门键（下图中的 1 0 号键），现在使用左侧的推杆可以控制小车的前后移动和转向（下图中的 1 + 1 - 摇杆）





根据这些按键编号，可以修改 `launch` 文件中的相关参数从而改变按键映射关系

花样玩法：

购买 ps3 手机支架，安卓手机装上 [小强图传 app](#)，这样可以实现图传遥控



教程(8)____ kinect1 代 ROS 驱动测试与安装

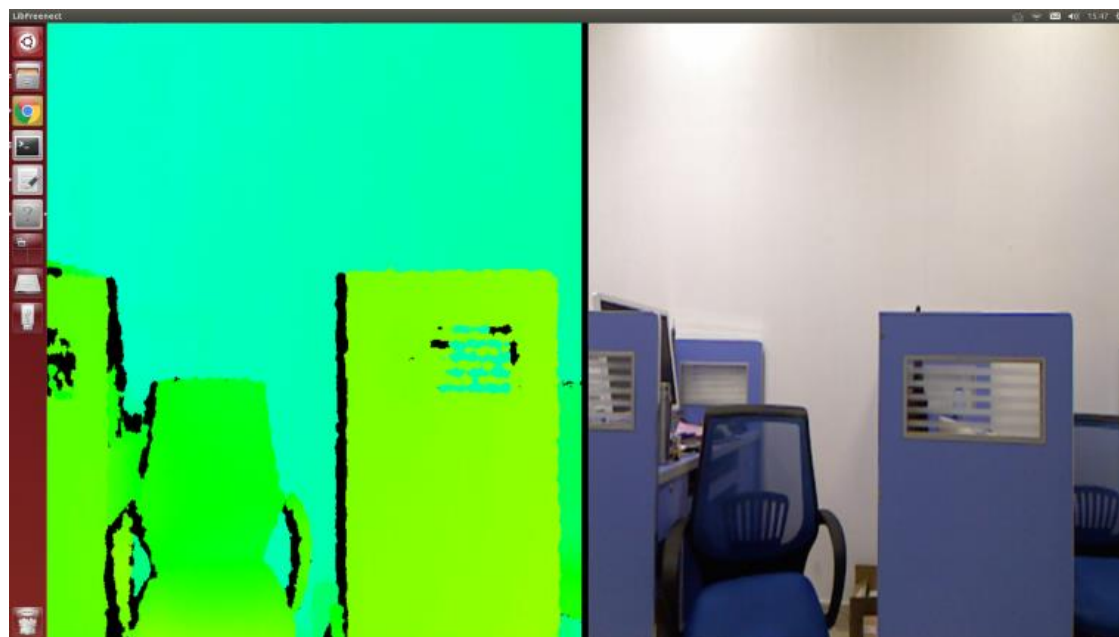
小强底盘输出一个 12v 电源 (DC 头, 贴有“kinect 供电”标签) 用于 kinect 供电。

1.LIBFRENECT 测试

将小强主机接入显示器和键盘, 在小强主机上新开一个命令终端输入

```
freenect-glvie
```

可以看到如下图的类似界面



2. ROS 驱动测试

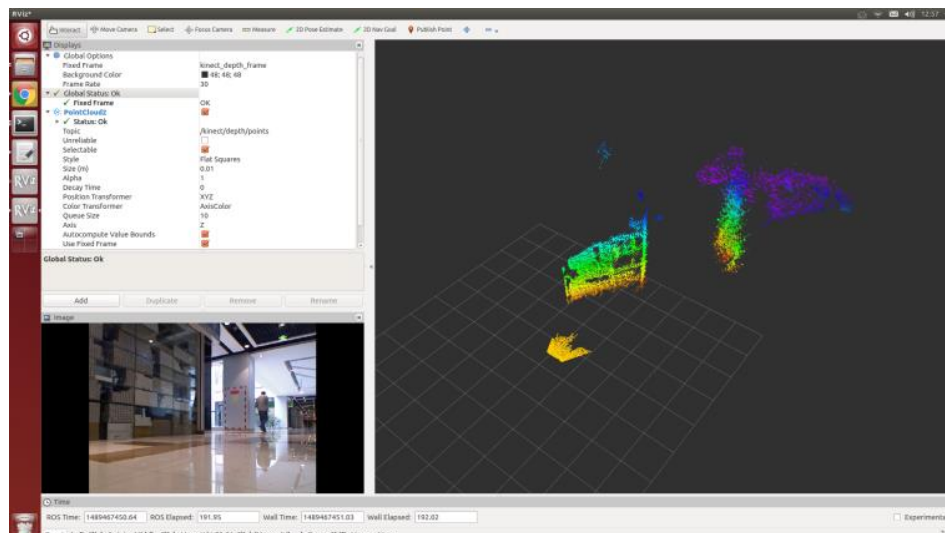
关闭步骤 1 中的程序, 新开一个命令窗口, 使用 `freenect_launch` 启动相关 kinect 节点

```
roslaunch freenect_launch freenect-xyz.launch
```

新开 1 个窗口打开 `rviz`

```
rviz
```

选择需要显示的内容, 例如 kinect 的 rgb 图像和深度点云, 显示效果如下



kinect 各项功能的开启在

/home/xiaoqiang/Documents/ros/src/freenect_stack/freenect_launch/launch/examples/freenect-xyz.launch 里面

```
<launch>
  <include file="$(find freenect_launch)/launch/freenect.launch">
    <arg name="camera" value="kinect" />
    <arg name="motor_processing" value="true" />
    <arg name="audio_processing" value="false" />
    <arg name="rgb_processing" value="true" />
    <arg name="ir_processing" value="false" />
    <arg name="depth_processing" value="true" />
    <arg name="depth_registered_processing" value="false" />
    <arg name="disparity_processing" value="false" />
    <arg name="disparity_registered_processing" value="false" />
    <arg name="num_worker_threads" value="4" />
  </include>
</launch>
```

通过设置 true 或者 false 来开启、关闭相应功能

3. 下文将介绍 KIENCT1 代的 ROS 驱动安装步骤

2016 年 7 月以后购买的用户不需要安装驱动，小强主机已经配置好 kinect 1 代驱动，本节教程已结束。

需要安装三个软件：

a.libfreenect

b.rgbd_launch

c.freenect_stack

A. LIBFRENECT

先将 kinect 接入小强主机，然后打开一个命令行终端，输入下列代码

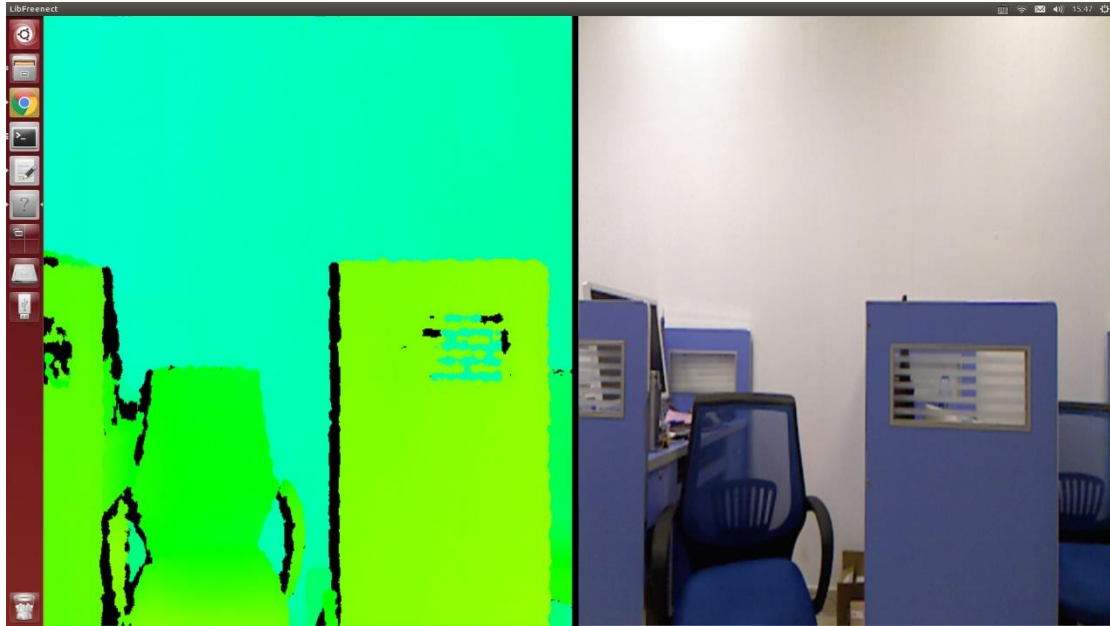
```
cd Documents
sudo apt-get install git-core cmake freeglut3-dev pkg-config build-essential libxmu-dev libxi-dev libusb-1.0-0-dev
git clone git://github.com/OpenKinect/libfreenect.git
cd libfreenect
mkdir build
cd build
//重点来了，下面配置将使能 kinect 音频和解决安装路径问题
```

```

cmake .. -DCMAKE_INSTALL_RPATH:STRING="/usr/local/bin;/usr/local/li
b" -DBUILD_REDIST_PACKAGE=OFF
make
sudo make install
sudo ldconfig /usr/local/lib64/
sudo freenect-glfwview

```

现在应该可以看到 kinect 的输出图像了



再进行外设权限配置操作

```

sudo adduser $xiaoqiang video //请将 xiaoqiang 换成自己电脑的账户名

```

增加一个 udev 规则,先打开 51-kinect.rules 文件

```

sudo gedit /etc/udev/rules.d/51-kinect.rules

```

拷贝如下内容后保存退出

```

# ATTR{product}=="Xbox NUI Motor"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02b0",
MODE="0666"
# ATTR{product}=="Xbox NUI Audio"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02ad",
MODE="0666"
# ATTR{product}=="Xbox NUI Camera"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02ae",
MODE="0666"
# ATTR{product}=="Xbox NUI Motor"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02c2",
MODE="0666"
# ATTR{product}=="Xbox NUI Motor"

```



```
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02be",  
MODE="0666"  
# ATTR{product}=="Xbox NUI Motor"  
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02bf",  
MODE="0666"
```

注销用户后重进系统，现在就能直接启用 kinect 了，不用 sudo 了

freenect-glfw

B. 安装 RGBD_LAUNCH

rgb_launch 包含了驱动安装包 openni_launch 或 freenect_launch 需要的通用 launch 文件。主要有两个重要的 launch 文件：

(1) processing.launch.xml：安装一系列 nodelets 去处理来自 RGB-D driver (openni_camera or freenect_camera) 的数据，还可以设定参数简化处理 nodelets 图像。

(2) kinect_frames.launch：为 Kinect 安装 tf tree。也可以从 openni_launch or freenect_launch 内部启动该文件。

rgb_launch 文件包含多个分散处理的 launch 文件。但只有 processing.launch.xml 可以在外部修改使用。

```
cd ~/Documents/ros/src  
git clone https://github.com/ros-drivers/rgb_launch.git  
cd ..  
catkin_make
```

C. 安装 FRENECT_STACK

```
cd ~/Documents/ros/src  
git clone https://github.com/ros-drivers/freenect_stack.git  
cd ..  
catkin_make
```

D. 驱动安装完成，现在可以在 ROS 中使用 KINECT 了，例如在 RVIZ 中观看 KINECT 输出的点云，参考本节开头的步骤 2

教程(9) 使用 rostopic 控制 kinect 的俯仰角度

准备工作:

请查看 kinect 版本, 在 kinect 底座标签上有注明。对于部分 model1473 的用户, 因为驱动的缺陷 (不影响 kinect 其它功能, 只涉及电机), 需要先进行如下操作, model1414 用户可以直接跳过。

ssh 登入小车主机

```
ssh xiaoqiang@192.168.0.xxx -X
```

freenect-micview

如果出现下图, 关闭上述命令, 继续教程

```
Reading reply: 00 E0 6F 0A 1D 00 00 00 00 00 00 00
Firmware successfully uploaded and launched. Device will disconnect and reenum
rate.
This is the libfreenect microphone waveform viewer. Press 'q' to quit or spaceba
ar to pause/unpause the view.
```

操作步骤:

1. 在本地虚拟机新开一个窗口, 启动 freenect_stack 驱动,

ssh 登入小车主机

```
ssh xiaoqiang@192.168.0.xxx -X
```

```
roslaunch freenect_launch freenect-xyz.launch
```

正常启动会出现下图, 如果出现红色错误 (驱动缺陷), 请 ctrl+c 关闭命令后等待 6 秒 (真的需要 6 秒), 再次尝试允许上面的 roslaunch 命令。

```
[ INFO] [1477406854.865362860]: Number devices connected: 1
[ INFO] [1477406854.865453964]: 1. device on bus 000:00 is a Xbox NUI Camera (2
e) from Microsoft (45e) with serial id 'A70774703536333A'
[ INFO] [1477406854.866242209]: Searching for device with index = 1
[ INFO] [1477406855.502427696]: flushDevice

[ INFO] [1477406855.502642152]: Starting a 3s RGB and Depth stream flush.
[ INFO] [1477406855.503054171]: Opened 'Xbox NUI Camera' on bus 0:0 with serial
number 'A70774703536333A'
[ WARN] [1477406856.952469775]: Could not find any compatible depth output mode
for 1. Falling back to default depth output mode 1.
[ INFO] [1477406856.961696739]: rgb_frame_id = 'kinect_rgb_optical_frame'
[ INFO] [1477406856.961735786]: depth_frame_id = 'kinect_depth_optical_frame'
[ WARN] [1477406856.975615344]: Camera calibration file /home/xiaoqiang/.ros/ca
mera_info/rgb_A70774703536333A.yaml not found.
[ WARN] [1477406856.975681775]: Using default parameters for RGB camera calibra
tion.
[ WARN] [1477406856.975777743]: Camera calibration file /home/xiaoqiang/.ros/ca
mera_info/depth_A70774703536333A.yaml not found.
[ WARN] [1477406856.975820217]: Using default parameters for IR camera calibrat
ion.
[ INFO] [1477406859.826134535]: Stopping device RGB and Depth stream flush.
```

2.在本地虚拟机新开一个窗口，发布电机角度控制命令

```
ssh 登入小车主机
```

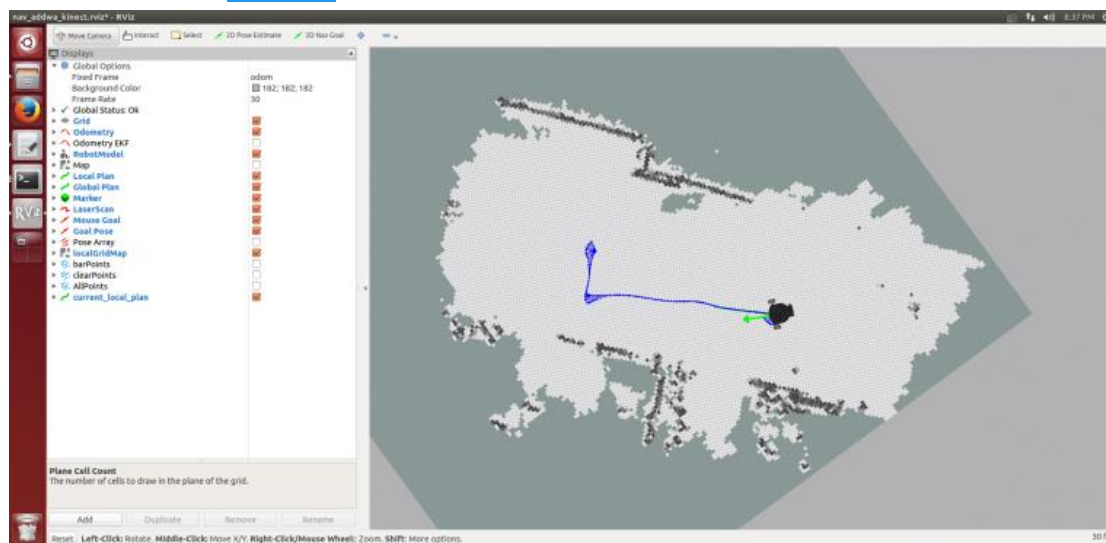
```
ssh xiaoqiang@192.168.0.xxx -X
```

```
rostopic pub /set_tilt_degree std_msgs/Int16 '{data: -20}' -r 1
```

如果一切正常，现在可以看到 kinect 的仰角不断变小，上述命令中的{data: -20}数字就代表角度，可以设置为 30 到-30 之间的整数

教程(10) 使用 kinect 进行自主移动避障

先看最后效果图，[点击观看](#)演示视频



原理:

freemove_stack 包提供 kinect 驱动，其发布的点云通过 image_pipeline 转换成障碍物栅格分布图。nav_test 软件包启动底盘导航程序后会自动处理分析障碍物分布图，之后根据 rviz 发布的目标导航点自主移动。

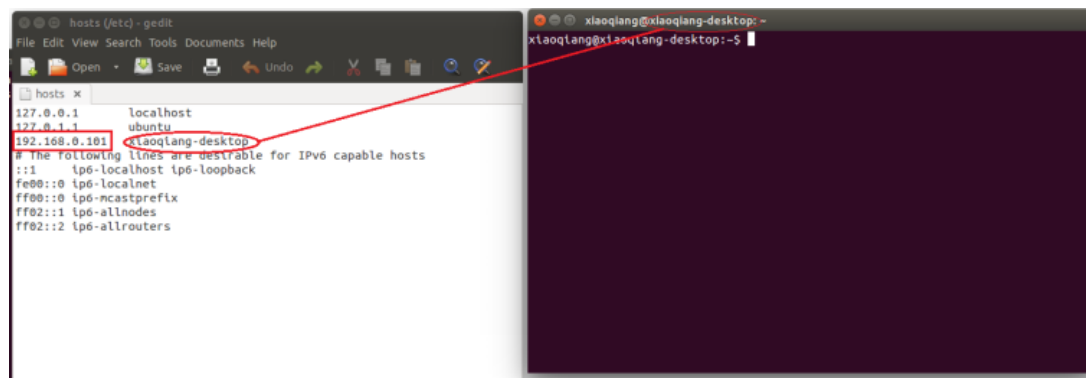
操作步骤:

1. 配置小车主机和本地虚拟机的 hosts 文件，使两者能互相访问对应的 ros 数据

1.A 配置本地虚拟机

```
sudo gedit /etc/hosts
```

将下图中的 ip 地址换成小车实际数值



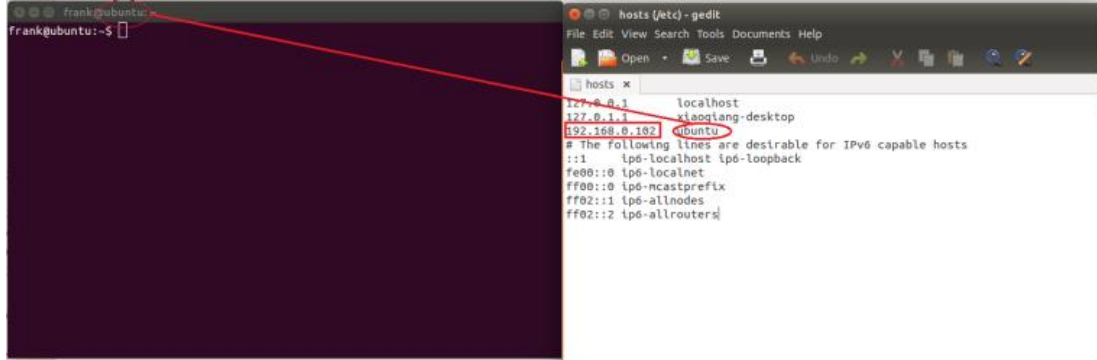
1.B 配置小车主机

局域网 `ssh` 登入小车主机

```
ssh xiaoqiang@192.168.0.101 -X
```

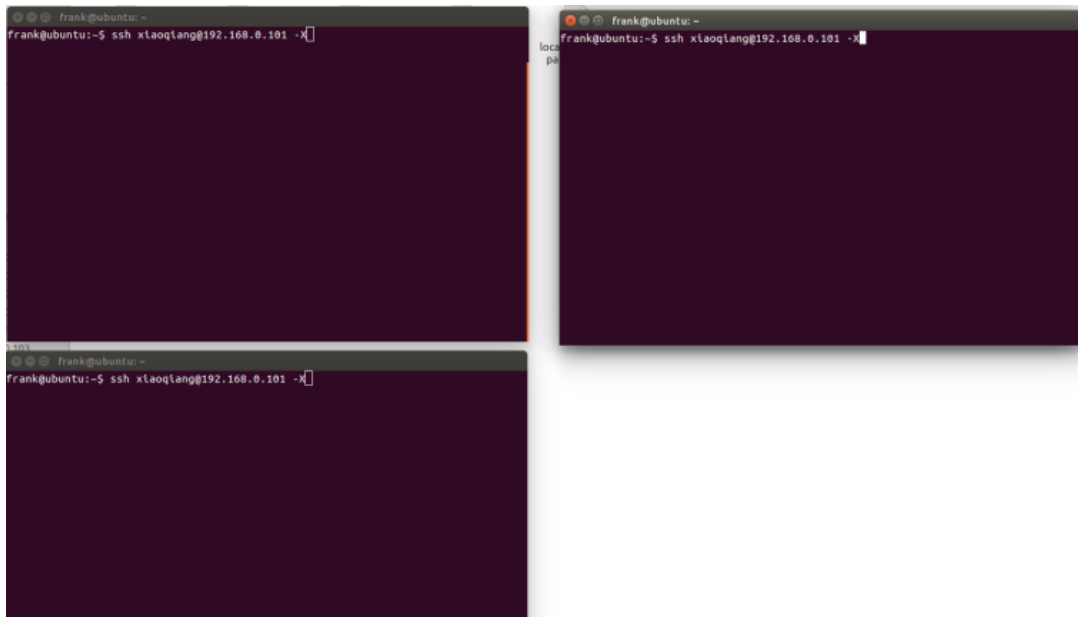
```
sudo gedit /etc/hosts
```

将下图中的 ip 地址换成本地虚拟机实际数值，主机名称改为虚拟机名称



2. 在本地虚拟机中新开 3 个窗口，分别 `ssh` 登入小车，启动原理部分提及的相关软件包

2.A SSH 登入



2.B 在第一个窗口启动 KINECT 驱动

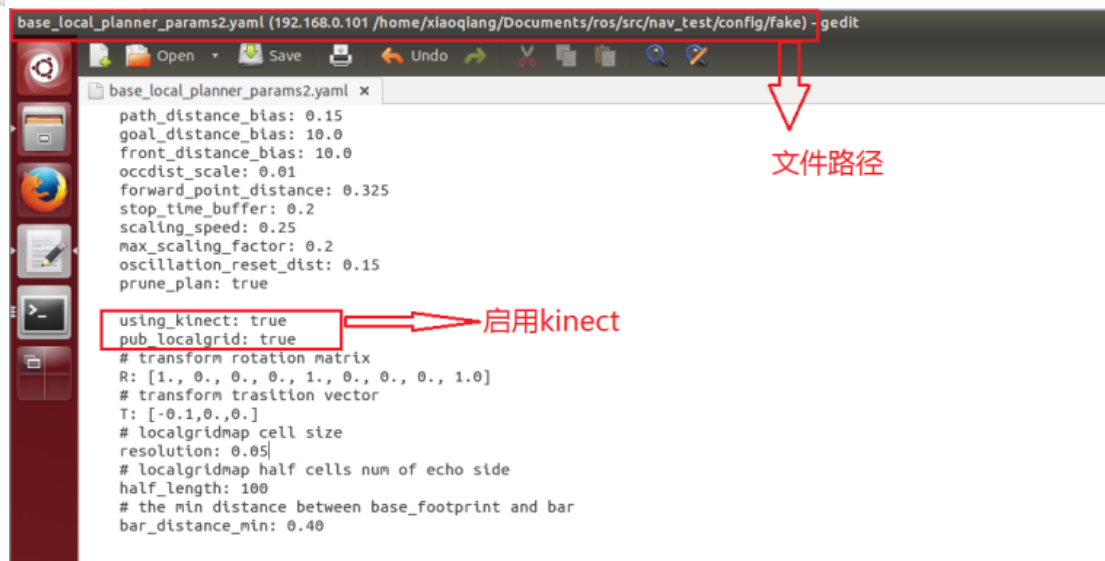
```
roslaunch freenect_launch kinect-xyz.launch
```

2.C 在第二个窗口设置 KINECT 俯仰角，这个角度不是任意的

```
rostopic pub /set_tilt_degree std_msgs/Int16 '{data: -19}' -r 1
```

2.D 编辑底盘导航程序配置文件

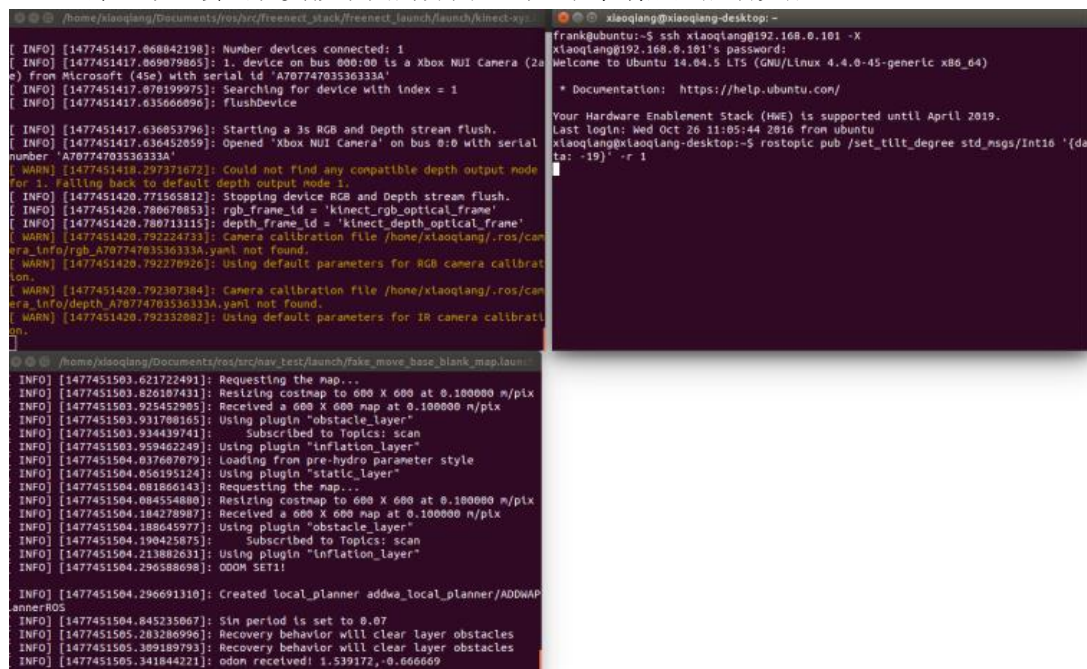
/HOME/XIAOQIANG/DOCUMENTS/ROS/SRC/NAV_TEST/CONFIG/FAKE/BASE_LOCAL_PLANNER_PARAMS2.YAML，使能 KINECT



2.E 在第三个窗口启动底盘导航程序

`roslaunch nav_test fake_move_base_blank_map.launch`

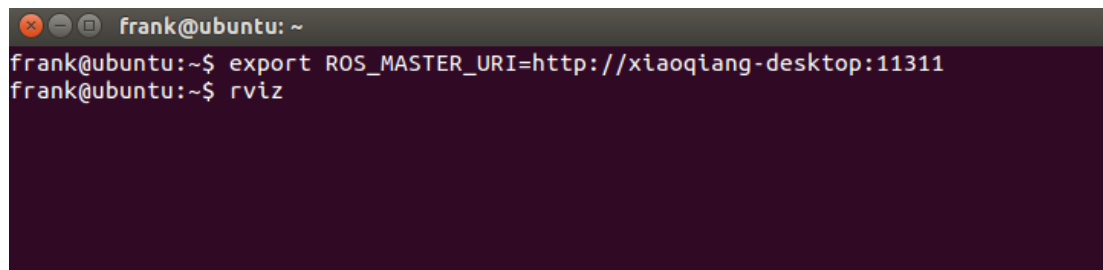
2.F 全部正常，会出现类似下图的界面，到此小车端配置启动完成



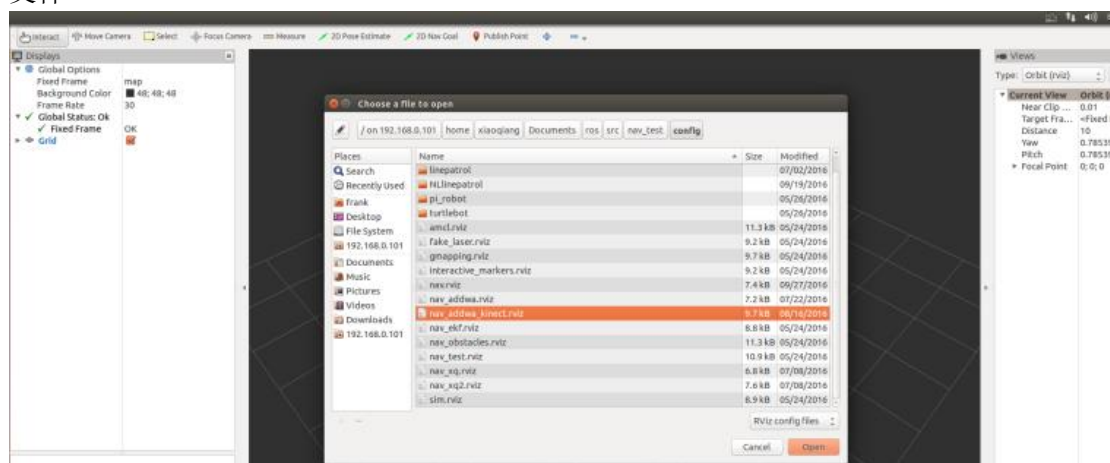
3. 在本地虚拟机中新开 1 个窗口，用来启动 rviz

3.A 加入 ROS 局域网后，打开 RVIZ

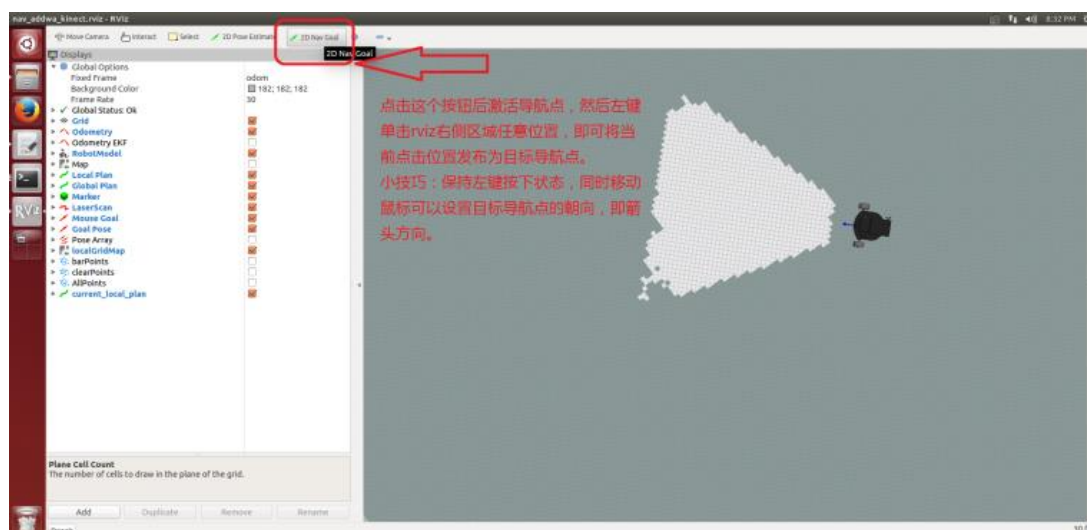
```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
rviz
```



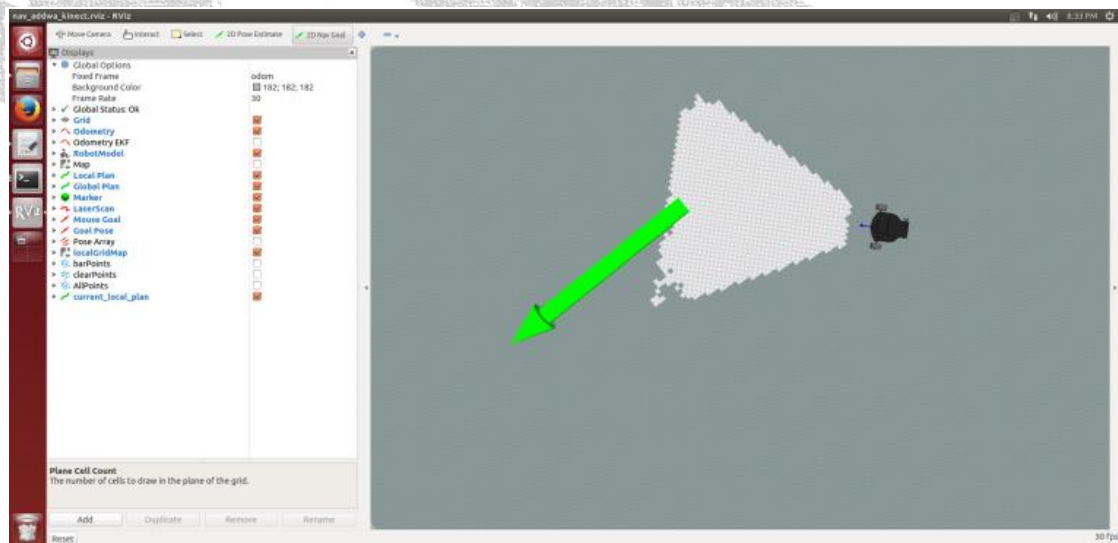
3.B 点击 RVIZ 界面左上角的 OPEN CONFIG，选择小车主机上的 /HOME/XIAOQIANG/DOCUMENTS/ROS/SRC/NAV_TEST/CONFIG/NAV_ADDWA_KINECT.RVIZ 配置文件



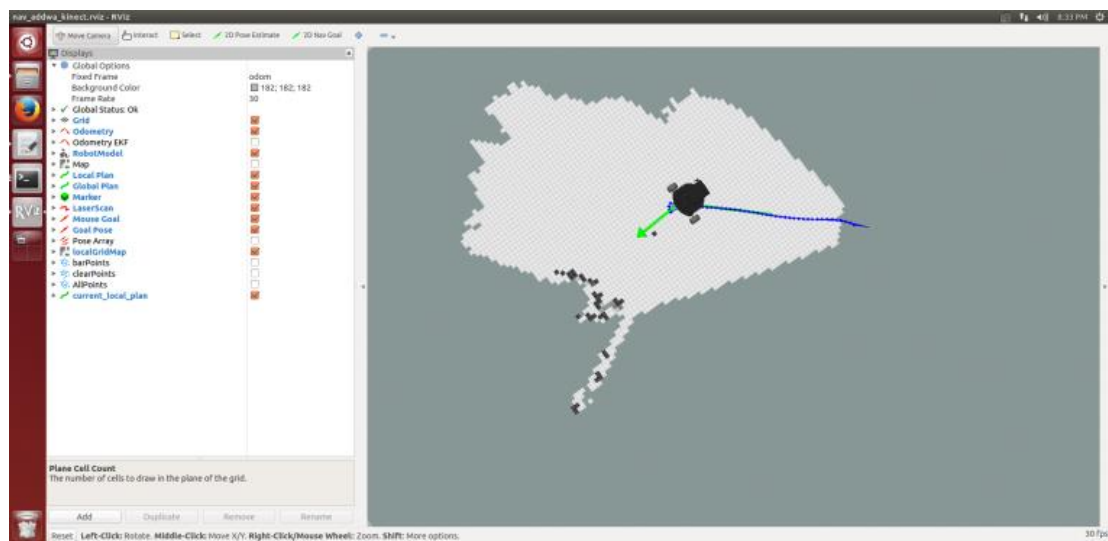
3.C 正常的话，现在 RVIZ 中将出现类似下图的画面，现在所有配置都已经完成，开始发布导航目标点



3.D 任性发布一个目标点，小车会开始自主移动



3.E 小车到达目标点，请继续尝试其它位置，本教程结束

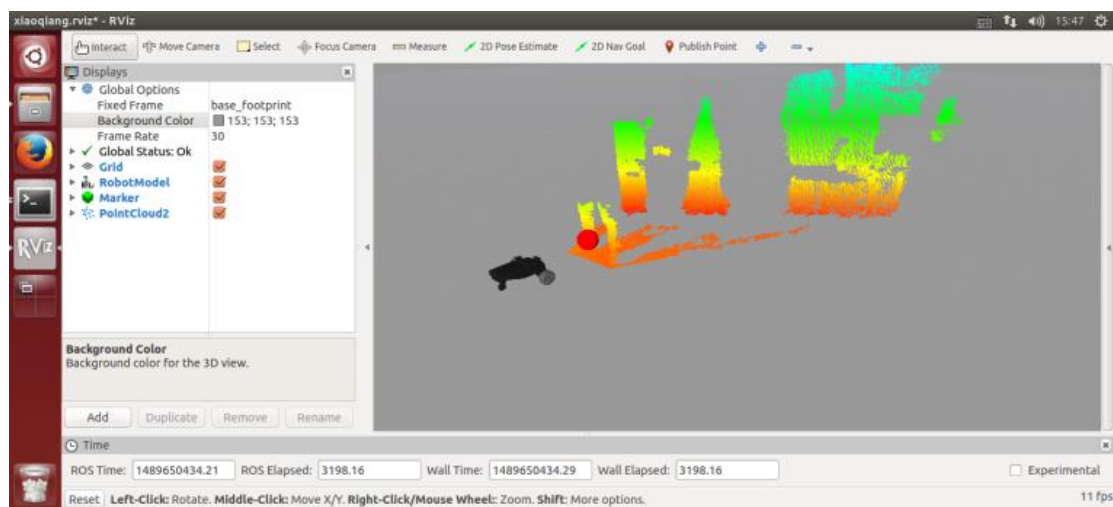


教程(11)___ kinect 跟随包 turtlebot_follower

注：2017 年 3 月 15 日之后购买的用户请跳过安装步骤，直接从步骤 2 开始测试

TURTLEBOT_FOLLOWER 利用深度摄像头反馈的点云图，计算一定区域内的点云中心坐标作为目标跟随点，根据这个坐标和设置的安全距离控制底盘移动实现跟随功能。

下图小强识别出人的双腿后，在人腿位置设置了一个红球作为目标点，[演示视频 1\(右键另存下载\)](#) ,[演示视频 2\(右键另存下载\)](#)



1. 安装软件包

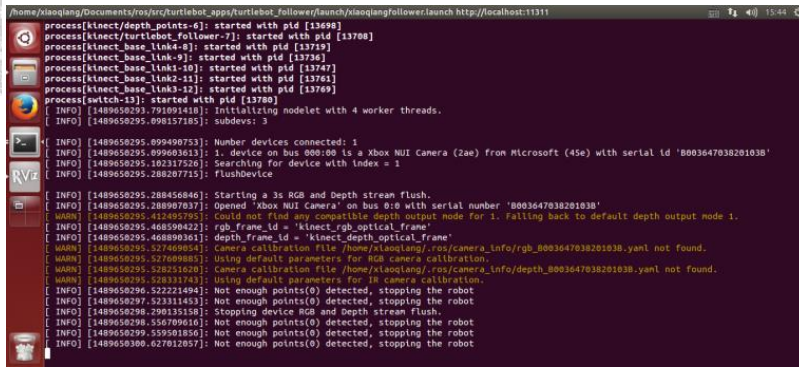
ssh 登入小强主机后，进入 ros 工作空间，下载安装测试软件包

```
ssh xiaoqiang@192.168.xxx.xxx
cd Documents/ros/src/
git clone https://github.com/turtlebot/turtlebot_msgs.git
git clone https://github.com/BlueWhaleRobot/turtlebot_apps.git
cd ..
catkin_make
```

2. 保证小强前方 2 米*2 米范围空旷无杂物，在小强主机上启动 TURTLEBOT_FOLLOWER 包

```
ssh xiaoqiang@192.168.xxx.xxx
roslaunch turtlebot_follower xiaoqiangfollower.launch
```

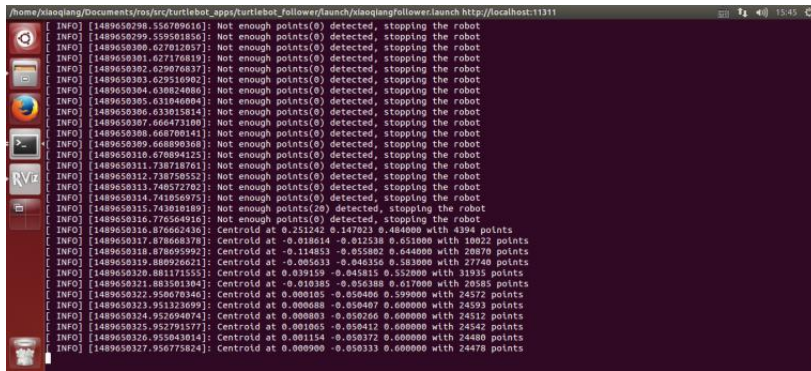
正常启动后会出现下图



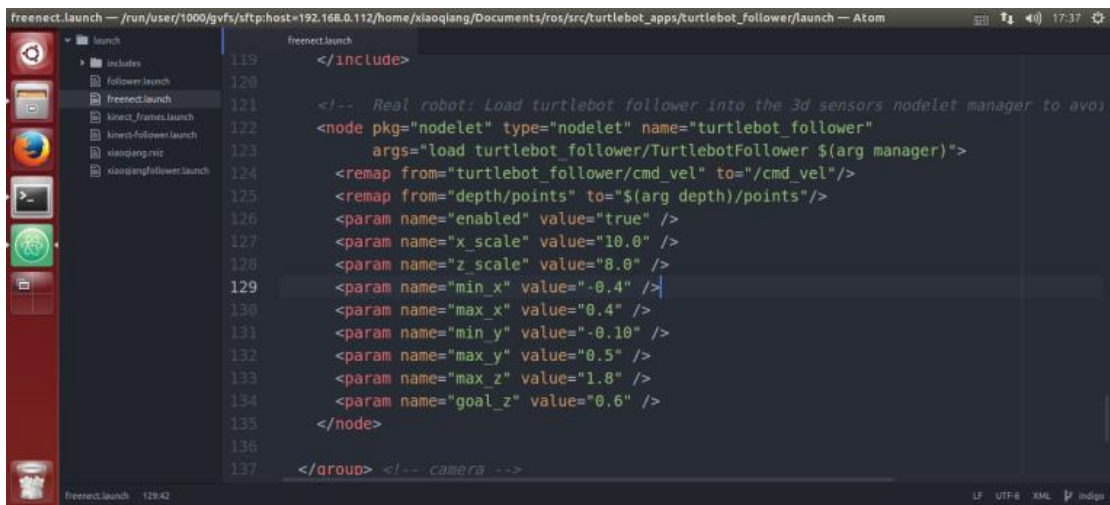
新开一个窗口，发布电机角度控制命令,保持 kienct 处于水平姿态

```
ssh xiaoqiang@192.168.0.xxx -X
rostopic pub /set_tilt_degree std_msgs/Int16 '{data: 0}' -r 1
```

3. 此时人进入小强前方视野，会激活小强的跟随功能，小强开始跟随人的移动而移动



修改小强主机上的 /home/xiaoqiang/Documents/ros/src/turtlebot_apps/turtlebot_follower/launch/freenect.launch 文件中如下图中的参数，可以控制跟随性能



教程(12) ROS 显示 kinect2 代的点云

本教程适用于 2016 年 12 月之后购买的小强用户，在此之前购买的用户请自行根据[这篇教程](#)安装 kinect2 代驱动。

1 启动 kinect2 代的 ROS 驱动

小强底盘输出一个 12v 电源 (DC 头，贴有“kinect 供电”标签) 用于 kinect 供电，kinect2 代需要插入小车主机蓝色 usb 3.0 接口。

将小强主机接入显示器和键盘，在小强主机上打开一个命令行终端输入

```
roslaunch kinect2_bridge kinect2-xyz.launch
```

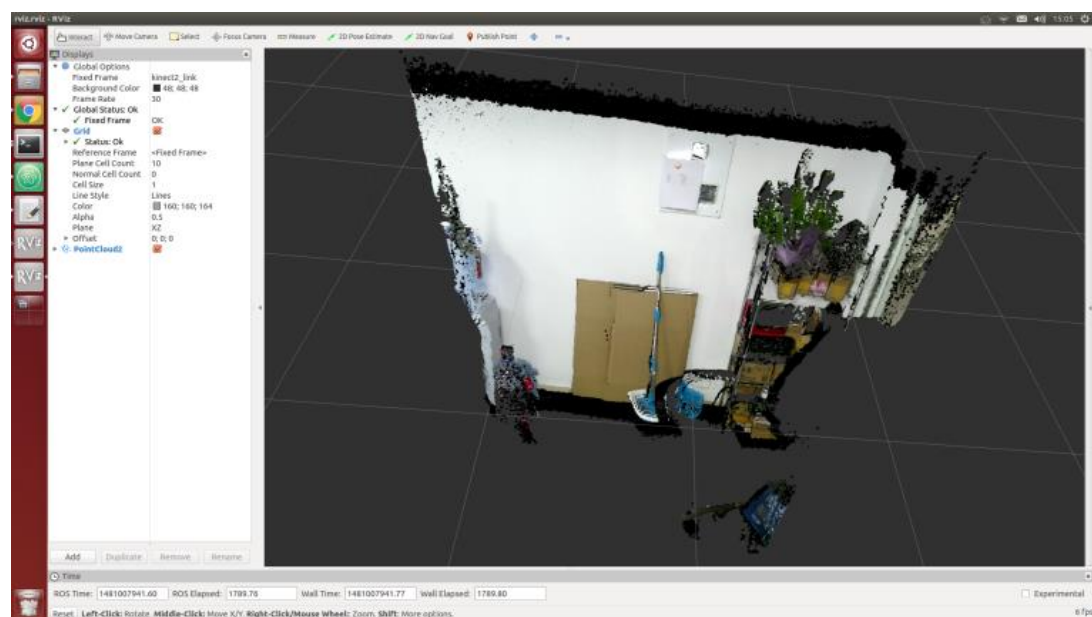
2 新开 1 个命令行终端，启动 rviz

Rviz

打开这个 rviz 配置文件

/home/xiaoqiang/Documents/ros/src/iai_kinect2/kinect2_bridge/launch/rviz.rviz, [点击可下载本文件](#)

一切正常的话，可以出现类似下图的界面



教程(13)___ rplidar 二代激光雷达的使用暨利用 udev 给小车增加串口设备

本节教程的 1、2、3、4 步骤仅用于演示给小强增加串口设备的方法，小强用户请接上硬件后直接跳到步骤 5 测试雷达。

小车主机和底盘的通信是通过串口实现的，在实际开发过程中我们可能会给小车增加串口外设，这会导致串口号（TTYUSB***）的混乱，引发小车底盘 ROS 驱动和串口设备的异常。下文将以 RPLIDAR 二代激光雷达为例，演示通过修改 UDEV 文件指定设备串口号的方式解决串口冲突问题。

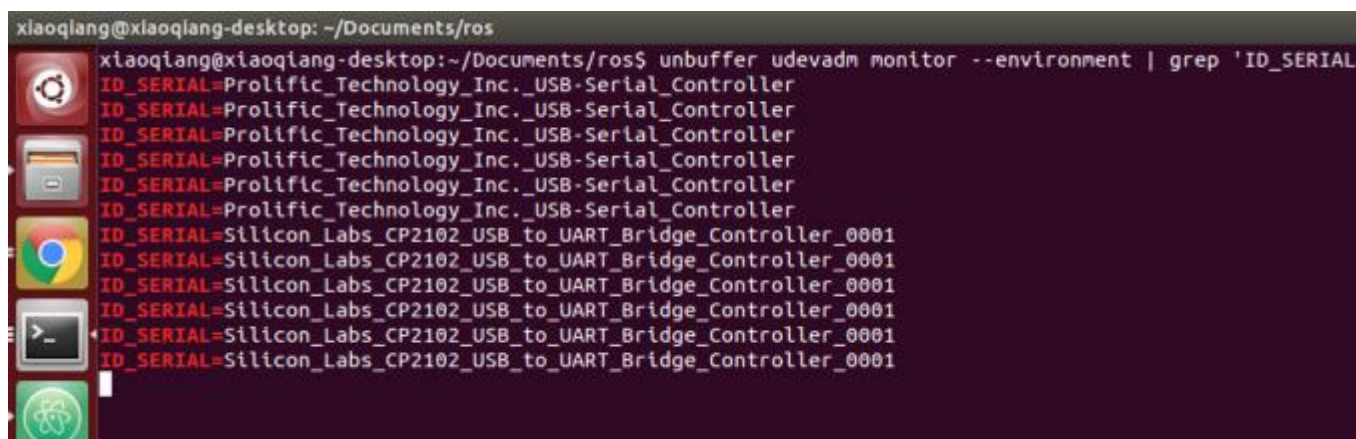
本文方法[出处](#)

1. 查看各个串口设备的 ID

```
sudo apt-get install expect-dev
unbuffer udevadm monitor --environment | grep 'ID_SERIAL='
```

将底盘通信 u 转串重新插拔一下，终端会打印出此设备的 ID 信息，例如下图："Prolific_Technology_Inc._USB-Serial_Controller"

再将激光雷达的 usb 适配器重新插入主机，终端也会打印出激光雷达的 ID 信息，例如下图："Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001"



```
xiaoqiang@xiaoqiang-desktop: ~/Documents/ros
xiaoqiang@xiaoqiang-desktop:~/Documents/ros$ unbuffer udevadm monitor --environment | grep 'ID_SERIAL=
ID_SERIAL=Prolific_Technology_Inc._USB-Serial_Controller
ID_SERIAL=Prolific_Technology_Inc._USB-Serial_Controller
ID_SERIAL=Prolific_Technology_Inc._USB-Serial_Controller
ID_SERIAL=Prolific_Technology_Inc._USB-Serial_Controller
ID_SERIAL=Prolific_Technology_Inc._USB-Serial_Controller
ID_SERIAL=Prolific_Technology_Inc._USB-Serial_Controller
ID_SERIAL=Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001
ID_SERIAL=Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001
ID_SERIAL=Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001
ID_SERIAL=Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001
ID_SERIAL=Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001
ID_SERIAL=Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001
```

2. 根据获取的串口设备的 ID，建立 UDEV 规则文件，将底盘通信 U 转串指定为 TTYUSB001，将激光雷达指定为 TTYUSB002

```
sudo gedit /etc/udev/rules.d/60-persistent-serial.rules
```

输入下面内容后保存，请将文中 ID_SERIAL 后面的字符串换成步骤 1 中获取的 ID

```
ACTION!="add", GOTO="persistent_serial_end"
SUBSYSTEM!="tty", GOTO="persistent_serial_end"
KERNEL!="ttyUSB[0-9]*", GOTO="persistent_serial_end"
```

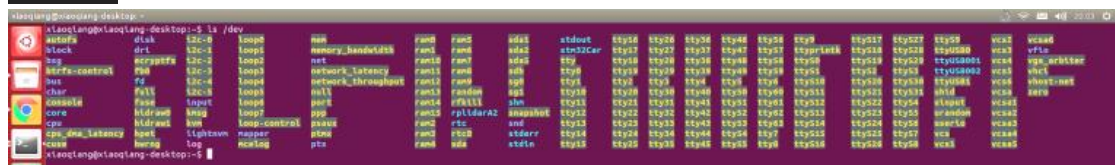
```
# This is old 11.10 style: IMPORT="usb_id --export %p"
IMPORT(builtin)="path_id"
ENV{ID_SERIAL}="Prolific_Technology_Inc._USB_Serial_Controller", SYMLINK="stm32Car", SYMLINK+="ttyUSB001", OWNER="xiaoqiang"
ENV{ID_SERIAL}="Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001", SYMLINK="rplidarA2", SYMLINK+="ttyUSB002", OWNER="xiaoqiang"
LABEL="persistent_serial_end"
```

更新系统 udev 规则

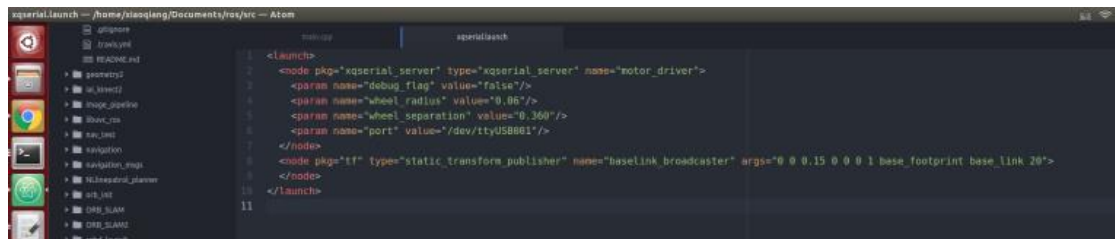
```
sudo udevadm control -reload
```

重新插拔所有 usb 串口设备,现在底盘通信 u 转串成功被识别为 ttyUSB001、激光雷达被识别为 ttyUSB002, 与设备插入顺序和端口无关。

ls /dev



3. 修改小车底盘 ROS 驱动节点 LAUNCH 文件, 将通信设备指定为上文设置的 TTYUSB001



4. 修改 RPLIDAR 二代激光的 ROS 驱动节点 LAUNCH 文件, 将通信设备指定为上文设置的 TTYUSB002

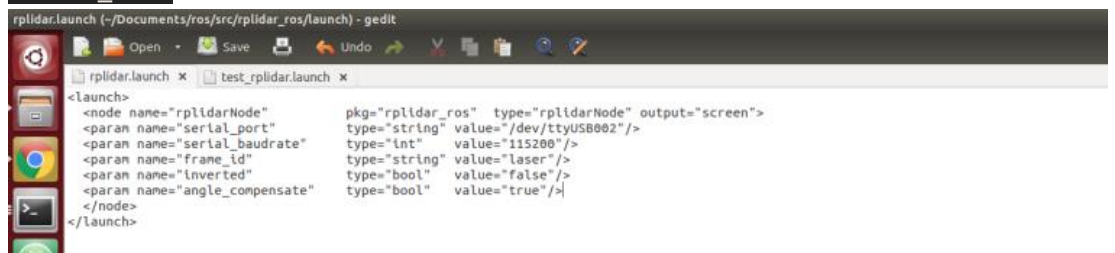
#驱动的安装

```
cd ~/Documents/ros/src
```

```
git clone https://github.com/BlueWhaleRobot/rplidar_ros.git
```

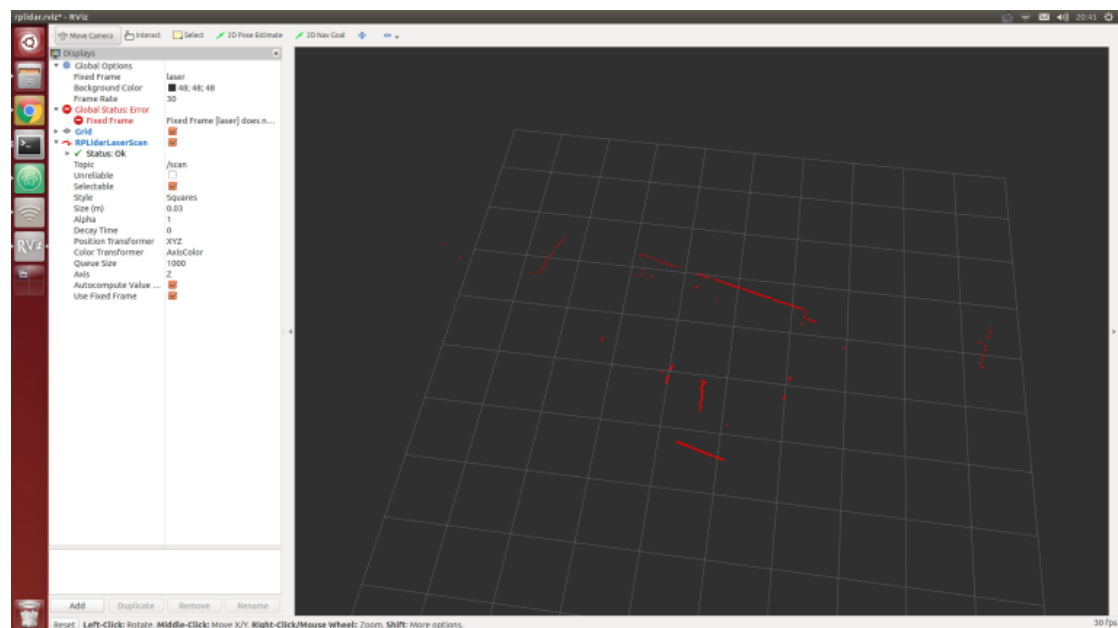
```
cd ..
```

```
catkin make
```



5. 重启小车，现在已经可以同时正常使用激光雷达和小车底盘，例如运行下述命令测试激光雷达

```
roslaunch rplidar_ros view_rplidar.launch
```



教程(14)___在 gmapping 下使用激光雷达 rplidar a2 进行建图

1、安装 GMAPPING, 2017 年 3 月 3 日之后收到货的用户可以跳过这个安装步骤, 直接从步骤 2 开始

ssh 登录小强主机, 进入小强 ros 工作空间

```
ssh xiaoqiang@192.168.XXX.XXX
cd Documents/ros/src/
```

下载两个 ros 软件包 [gmapping](https://github.com/BlueWhaleRobot/slam_gmapping)、[openslam_gmapping](https://github.com/BlueWhaleRobot/openslam_gmapping)

```
git clone https://github.com/BlueWhaleRobot/slam_gmapping.git
git clone https://github.com/BlueWhaleRobot/openslam_gmapping.git
```

编译、完成安装

```
cd ..
catkin_make
```

2、启动 GMAPPING 节点

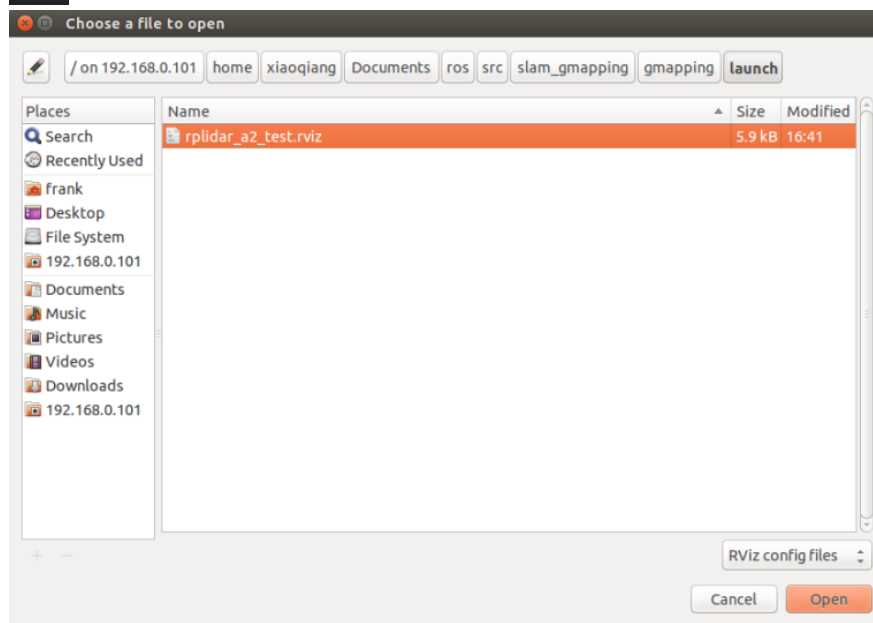
确保雷达安装正确, ssh 进入小强主机后启动 gmapping 中的 launch 文件

```
ssh xiaoqiang@192.168.XXX.XXX
roslaunch gmapping slam_gmapping_xiaoqiang_rplidar_a2.launch
```

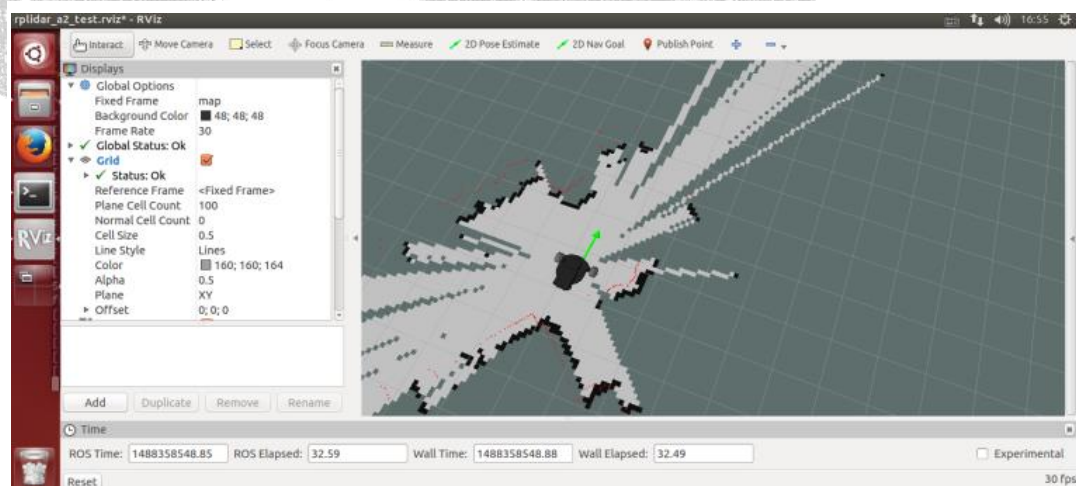
本地虚拟机打开 rviz, 选择打开小强 ros 工作目录下的

slam_gmapping/gmapping/launch/rplidar_a2_test.rviz 配置文件

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
rviz
```



等待几秒，正常情况会出现下图的类似结果



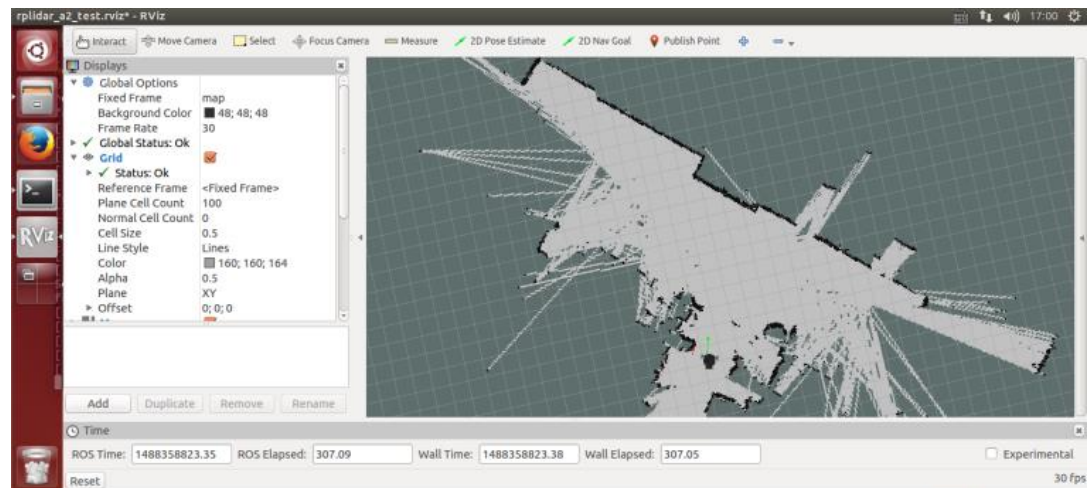
3、遥控小强运动开始建图

第一种方式，使用 windows 遥控端，参考这篇帖子[[小强图传遥控 windows 客户端](#)]

第二种方式，使用键盘遥控

```
ssh xiaoqiang@192.168.XXX.XXX
rosrun nav_test control.py
```

第三种方式，使用手机 app,参考这篇帖子[[小强手机遥控 app 安卓版](#)]

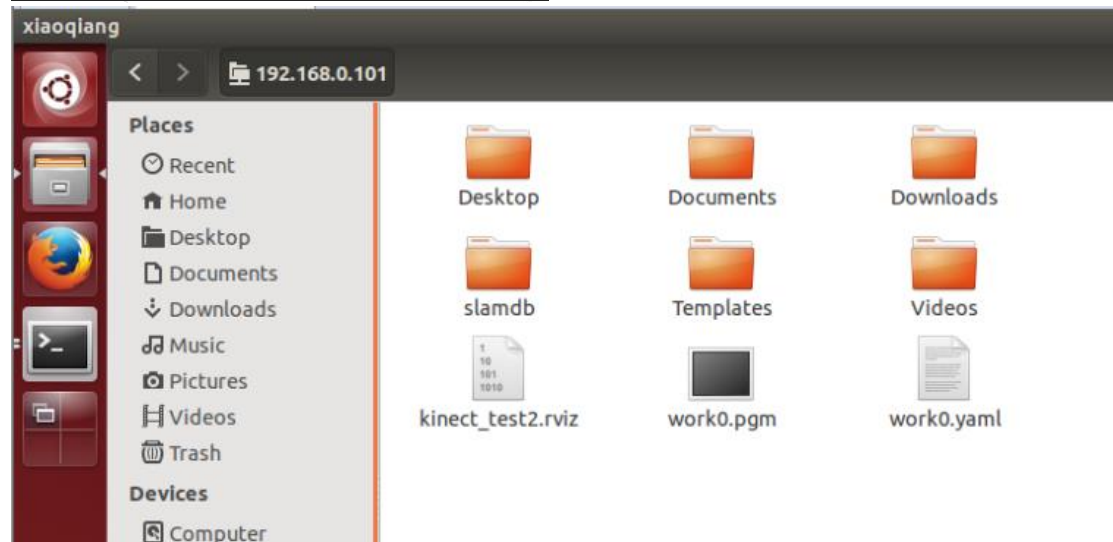


4、保存地图，本文结束

ssh 登录小强，在小强 home 目录下保存为 work0 开头的文件

```
ssh xiaoqiang@192.168.XXX.XXX
```

```
roslaunch map_server map_saver -f work0
```



教程(15) AMCL 导航测试

下文将演示 AMCL 导航操作, 使用 rplidar a2 作为 scan 输入, [教程 14](#) 中建立的地图文件作为全局 map

一、准备工作, 先安装升级 nav_test、laser_filters 软件包

1. SSH 登入小强主机, 进入小强 ROS 工作空间

```
ssh xiaoqiang@192.168.xxx.xxx -X
cd Documents/ros/src/
```

2. 更新升级软件包

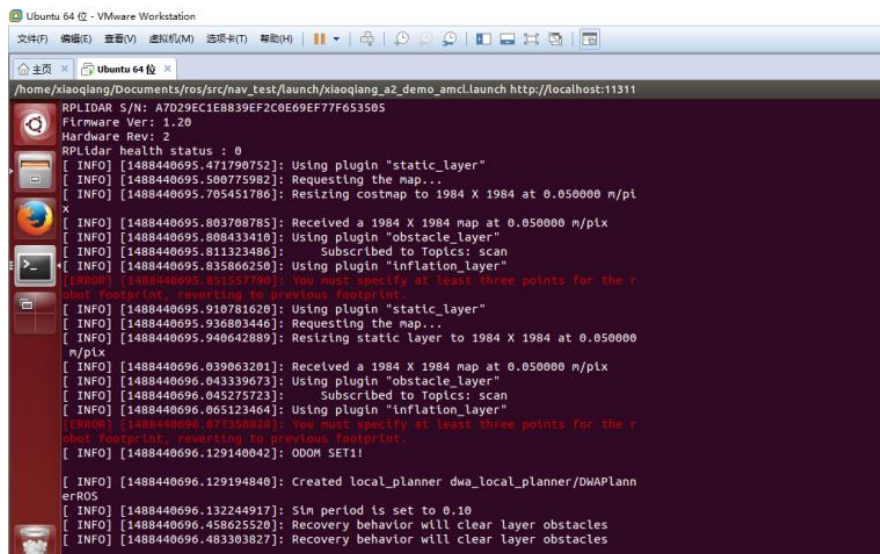
```
rm -rf laser_filters
git clone https://github.com/BlueWhaleRobot/laser_filters.git
cd nav_test
git stash
git pull
cd ..
cd ..
catkin_make
```

3. 更新小强 HOSTS 文件和本地虚拟机的 HOSTS 文件, 使小强和本地虚拟机可以相互通信, 参考教程 13 中的 1.A 和 1.B 部分

二、启动导航节点

```
roslaunch nav_test xiaoqiang_a2_demo_amcl.launch
```

正常会出现下图的类似结果, 同时雷达开始旋转

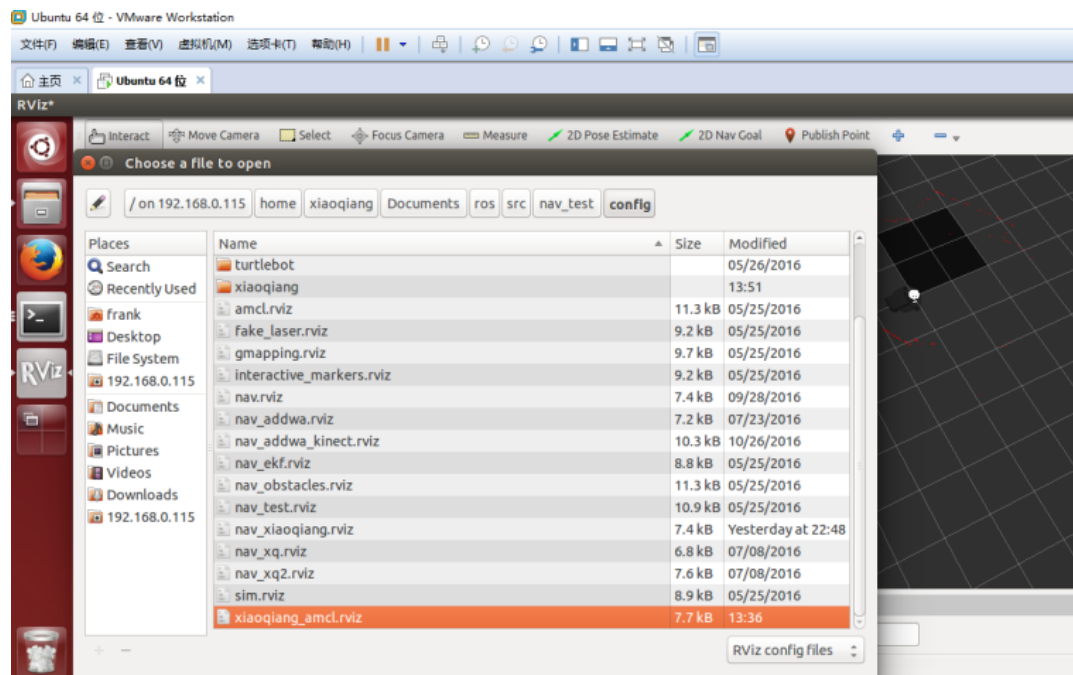


```
Ubuntu 64 位 - VMware Workstation
/home/xiaoqiang/Documents/ros/src/nav_test/launch/xiaoqiang_a2_demo_amcl.launch http://localhost:11311
RPLIDAR S/N: A7D29EC1E8839EF2C0E69EF77F653505
Firmware Ver: 1.20
Hardware Rev: 2
RPLidar health status : 0
[ INFO ] [1488440695.471790752]: Using plugin "static_layer"
[ INFO ] [1488440695.500775982]: Requesting the map...
[ INFO ] [1488440695.705451786]: Resizing costmap to 1984 X 1984 at 0.050000 m/plx
[ INFO ] [1488440695.803708785]: Received a 1984 X 1984 map at 0.050000 m/plx
[ INFO ] [1488440695.808433410]: Using plugin "obstacle_layer"
[ INFO ] [1488440695.811323486]: Subscribed to Topics: scan
[ INFO ] [1488440695.835866250]: Using plugin "inflation_layer"
[ERROR] [1488440698.821537790]: You must specify at least three points for the r
node Footprint, passing to GreenTeam Footprint.
[ INFO ] [1488440695.930781620]: Using plugin "static_layer"
[ INFO ] [1488440695.936883446]: Requesting the map...
[ INFO ] [1488440695.940642889]: Resizing static layer to 1984 X 1984 at 0.050000
m/plx
[ INFO ] [1488440696.039063201]: Received a 1984 X 1984 map at 0.050000 m/plx
[ INFO ] [1488440696.043339673]: Using plugin "obstacle_layer"
[ INFO ] [1488440696.045275723]: Subscribed to Topics: scan
[ INFO ] [1488440696.065123464]: Using plugin "inflation_layer"
[ERROR] [1488440698.877358028]: You must specify at least three points for the r
node Footprint, passing to GreenTeam Footprint.
[ INFO ] [1488440696.129140042]: ODOM SET!!
[ INFO ] [1488440696.129194840]: Created local_planner dwa_local_planner/DWAPlann
erROS
[ INFO ] [1488440696.132244917]: Sin period is set to 0.10
[ INFO ] [1488440696.458625520]: Recovery behavior will clear layer obstacles
[ INFO ] [1488440696.483303827]: Recovery behavior will clear layer obstacles
```

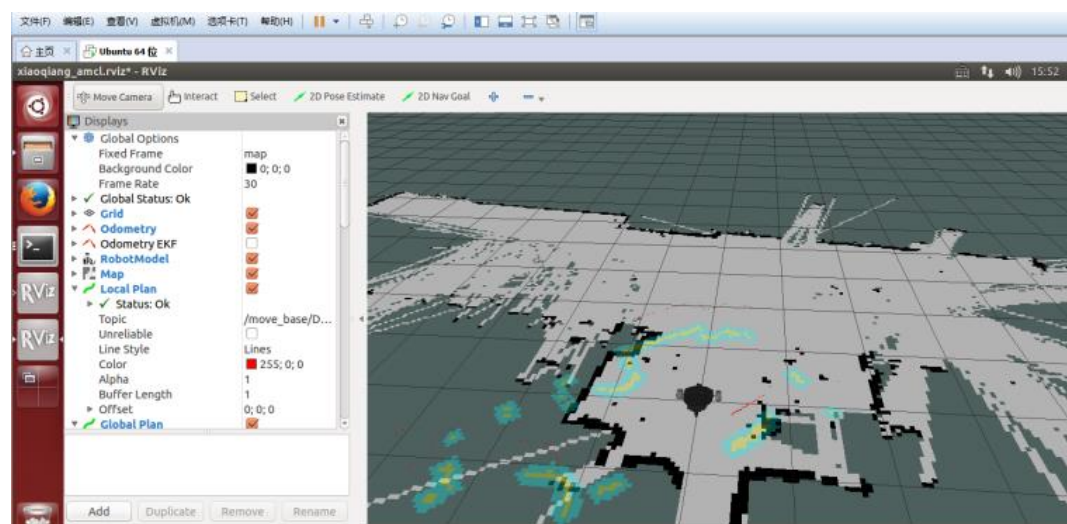
三、打开操作客户端

1. 在本地虚拟中启动 RVIZ, 选择打开小强 ROS 工作目录下的 NAV_TEST/CONFIG/XIAOQIANG_AMCL. RVIZ 配置文件

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
rviz
```



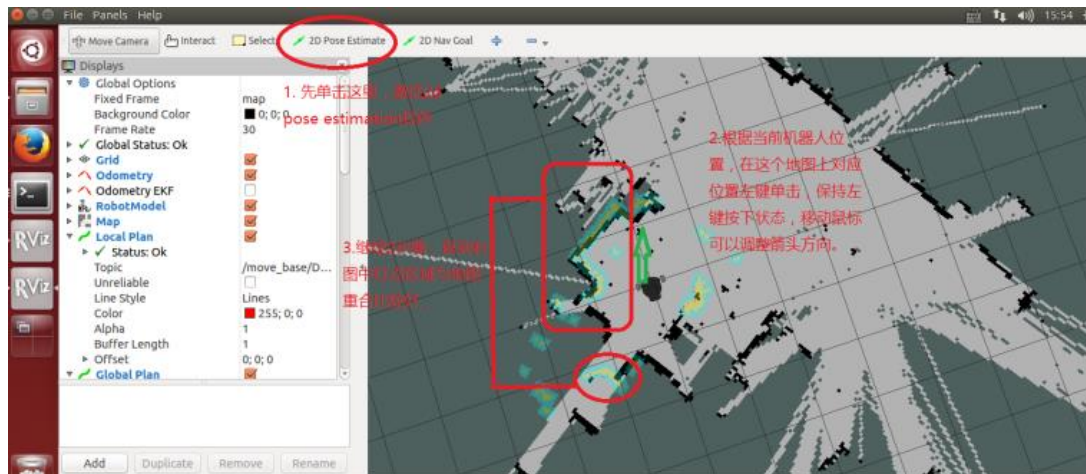
2. 等待几秒后, RVIZ 正常会出现类似下图的界面



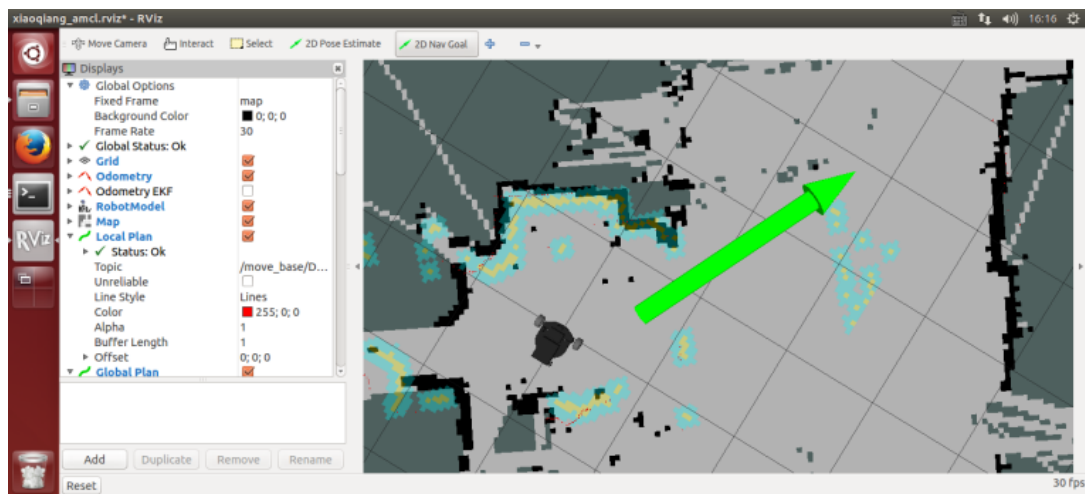
四、开始导航测试

1. 在 RVIZ 中使用 2D POSE ESTIMATION 设置机器人的初始 POSE 在 MAP 中的位置

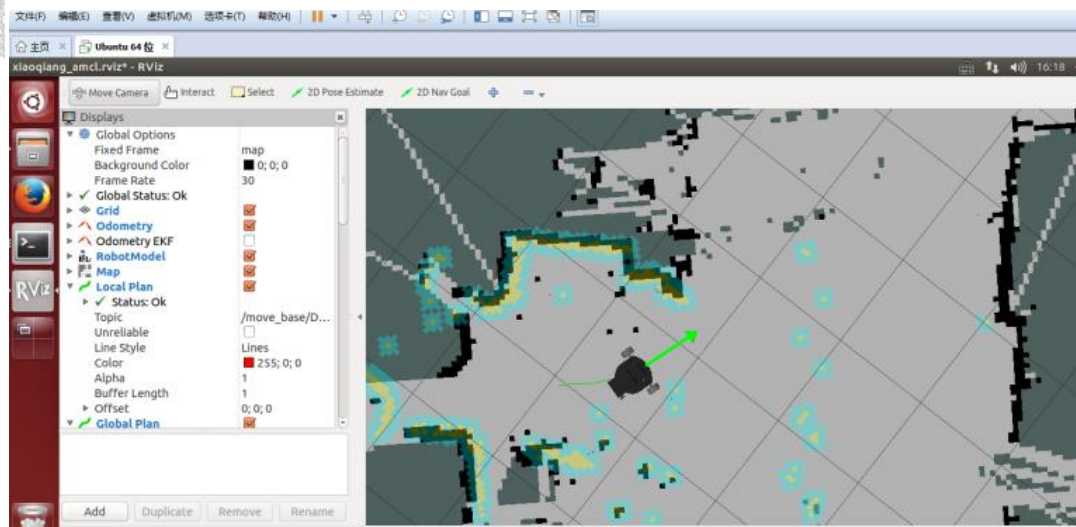
因为 AMCL 算法需要一个较为精确的初始值，才能进一步由当前雷达扫描点阵匹配出机器人在 MAP 中的真实位置。



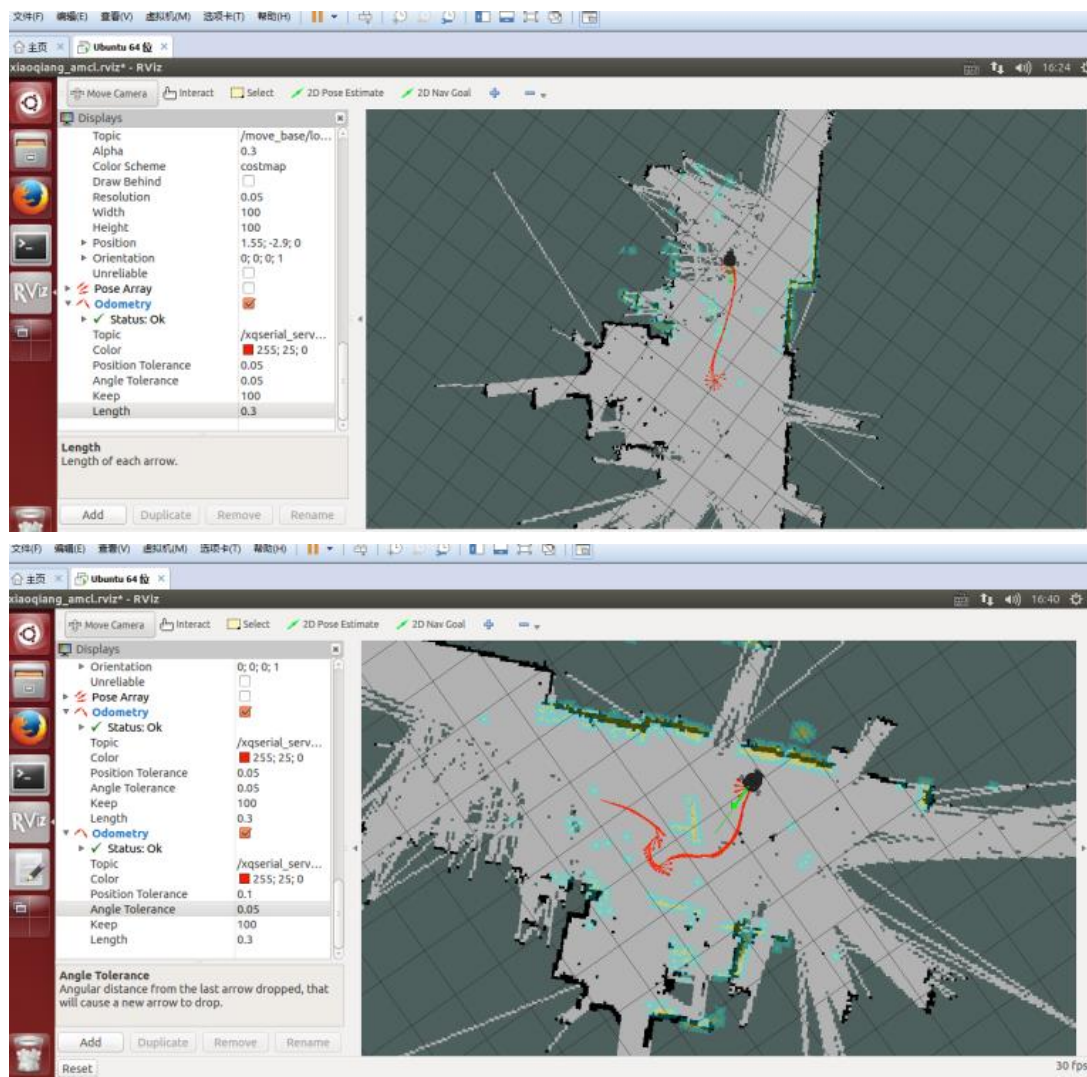
2. 在 RVIZ 中使用 2D NAV GOAL 给小强发布目标点



3. 小强开始自主移动到指定位置

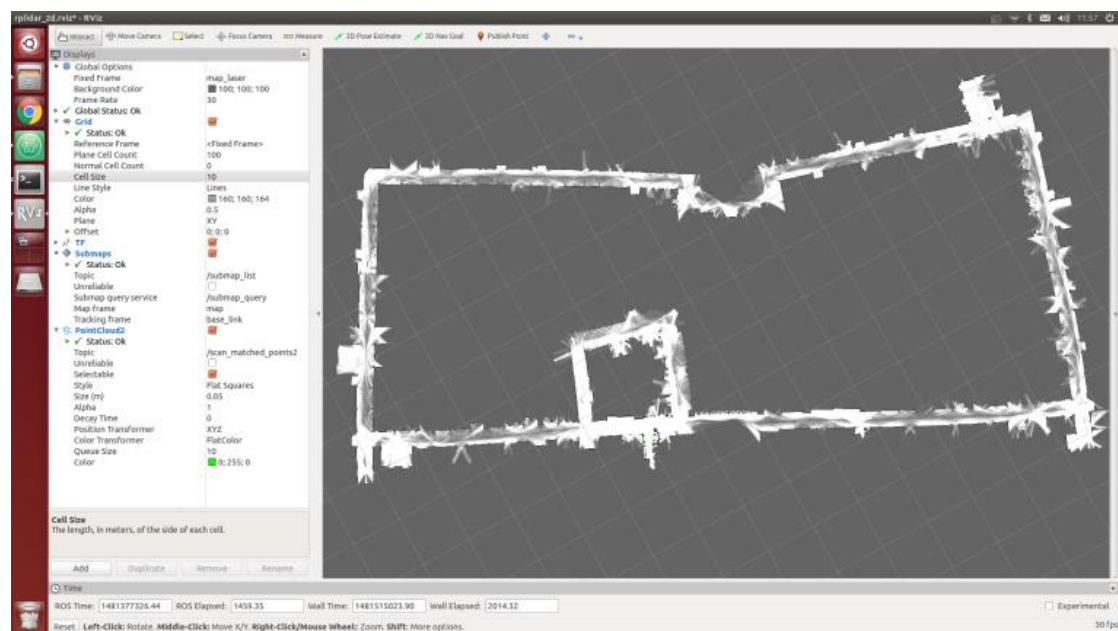


五、请自由设置小强的 2D Nav Goal, 观察小强的运动情况, 本文结束



教程(16)____大范围激光雷达 slam 与实时回路闭合测试

借助谷歌的 CARTOGRAPHER 配合 SLAMTEC 的激光雷达，我们可以尝试对大型建筑建立平面图。先看我们自己的 DEMO 演示效果，[点击观看视频](#)。在本 DEMO 中，小强实际运行在一个 5000 平米的写字楼走廊里，走廊两侧存在大量的玻璃幕墙，大楼中央存在一个大面积空旷地，加上 RPLIDAR 的测距范围只有 6 米，因此下图的最终效果还算理想（只使用激光雷达，没有开启 IMU 和底盘 ODOMETER，大回路路径仍然成功闭合）



本文操作思路：因为是大范围建图，WIFI 网络覆盖是一个问题，所以我们借助蓝牙手柄来遥控小车运动。期间通过 ROSBAG 录制激光雷达数据，手柄遥控小车在建图范围内跑一圈，最后重放 BAG 建图。

注：以下所有操作在小车主机 UBUNTU 上完成

A. 准备工作：

1. 安装 rplidar 驱动

对于 2016 年 1 月 15 日之后购买小强开发平台的用户，rplidar 驱动已经配置好。rplidar 的驱动安装请参考[这篇教程](#)

2. 安装 ps3 手柄驱动

对于 2016 年 1 月 15 日之后购买小强开发平台的用户，请跳过本步骤。请参考[这篇安装教程](#)

3. 安装 cartographer_ros

请参考这篇[安装教程](http://community.bwbot.org/topic/136/google)：<http://community.bwbot.org/topic/136/google> 激光雷达 slam 算法 cartographer 的安装及 bag 包 demo 测试

B. 操作步骤:

1. 新开一个窗口启动 rplidar

```
roslaunch rplidar_ros rplidar.launch
```

2. 新开两个窗口启动 ps3 手柄遥控程序, 按手柄连接键连上小车

第一个窗口

```
sudo bash
```

```
roslaunch ps3joy ps3joyfake_node.py
```

第二个窗口

```
roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
```

3. 新开一个窗口启动 rosbag 录制进程, 开始录制激光雷达数据 / scan

```
rosbag record /scan
```

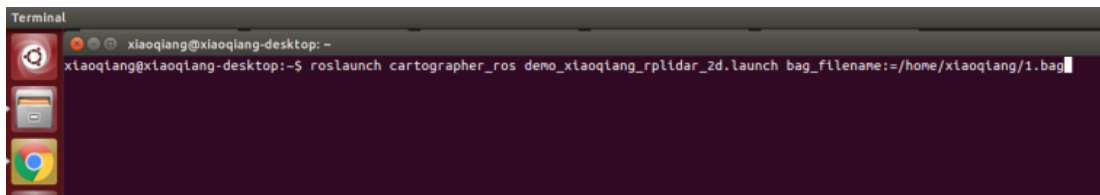
4. 用手柄遥控小车运动, 绕建图区域一圈, 也可以多圈

5. bag 录制完成, 关闭上文的 1、2、3 窗口

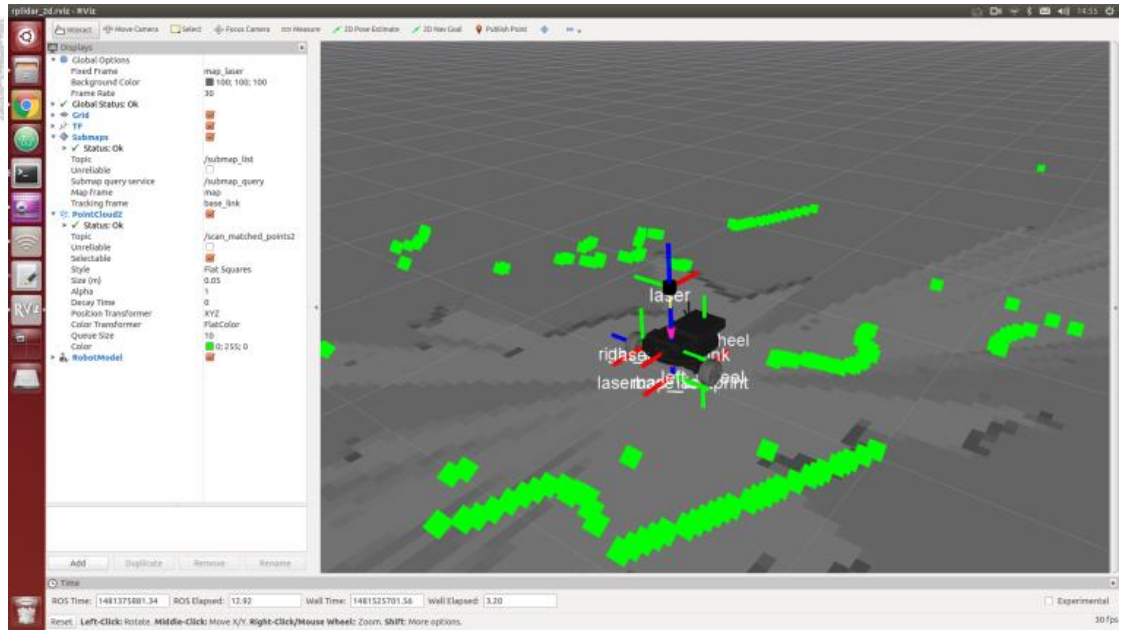
新录制的点 b a g 文件在小强 home 目录下, 将其重命名为 1 .bag

6. 启动 cartographer_ros 开始 bag 回放建图

```
roslaunch cartographer_ros demo_xiaoqiang_rplidar_2d.launch bag_filename:=/home/xiaoqiang/1.bag
```



7. 一切正常的话, 现在可以看到下图的类似效果, 等待 bag 包 play 完



8. 保存地图，本文结束

```
rosservice call /finish_trajectory "stem: 'rplidar_test'"
```


教程(17)___利用 ORB_SLAM2 建立环境三维模型

想要实现视觉导航，空间的三维模型是必须的。[ORB SLAM](#) 就是一个非常有效的建立空间模型的算法。这个算法基于 ORB 特征点的识别，具有准确度高，运行效率高的特点。我们在原有算法的基础上进行了修改，增加了地图的保存和载入功能，使其更加适用于实际的应用场景。下面就介绍一下具体的使用方法。

准备工作

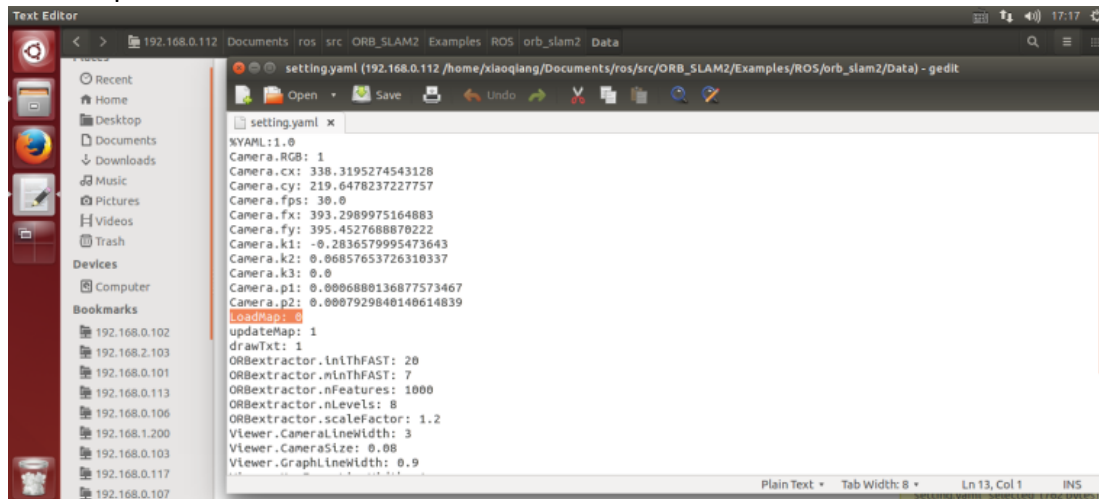
在启动 ORB_SLAM2 之前，请先确认小强的摄像头工作正常。

ORB_SLAM2 建图过程中需要移动小车，移动小车过程中 ORB_SLAM2 的运行状态不方便显示 (ssh 方式比较卡顿，也不可能拖着显示器)，因此请先安装好[小强图传遥控 windows 客户端](#)。

1. 启动 ORB_SLAM2

更改配置文件

ORB_SLAM2 的配置文件位于 `/home/xiaoqiang/Documents/ros/workspace/src/ORB_SLAM2/Examples/ROS/orb_slam2/Data` 文件夹内。更改 `setting.yaml` 其中的 `LoadMap` 的值，将其设置为 0。当设置为 1 的时候程序会在启动后自动从 `/home/xiaoqiang/slamdb` 文件夹内载入地图数据。当设置成 0 时，就不会载入地图数据。由于我们是要创建地图，所以 `LoadMap` 要设置为 0。



ssh 方式进入小强，执行以下指令

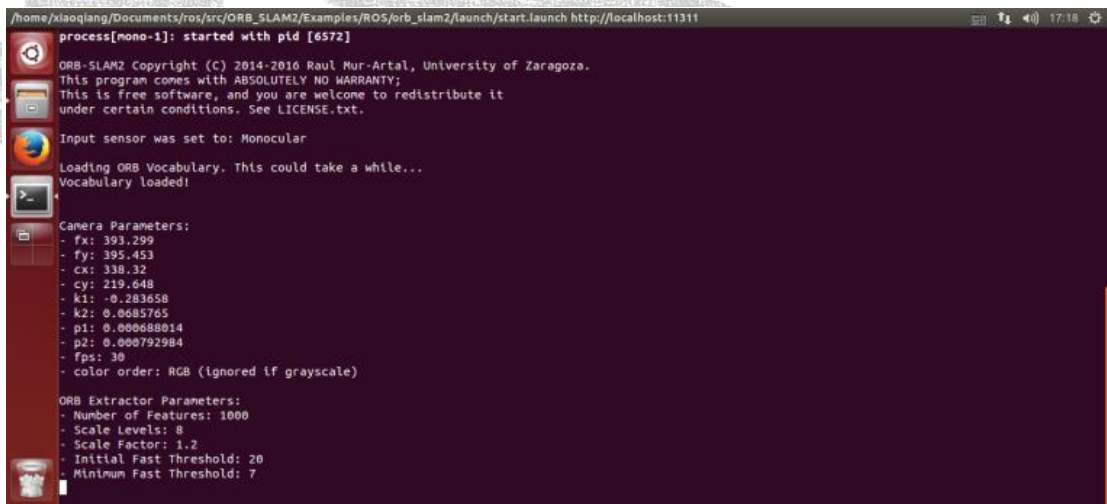
```
ssh -X xiaoqiang@192.168.xxx.xxx
```

#2017 年之前购买用户，请执行

```
roslaunch ORB_SLAM2 start.launch
```

#2017 年之后购买用户，请执行

```
roslaunch orb_slam2 start.launch
```

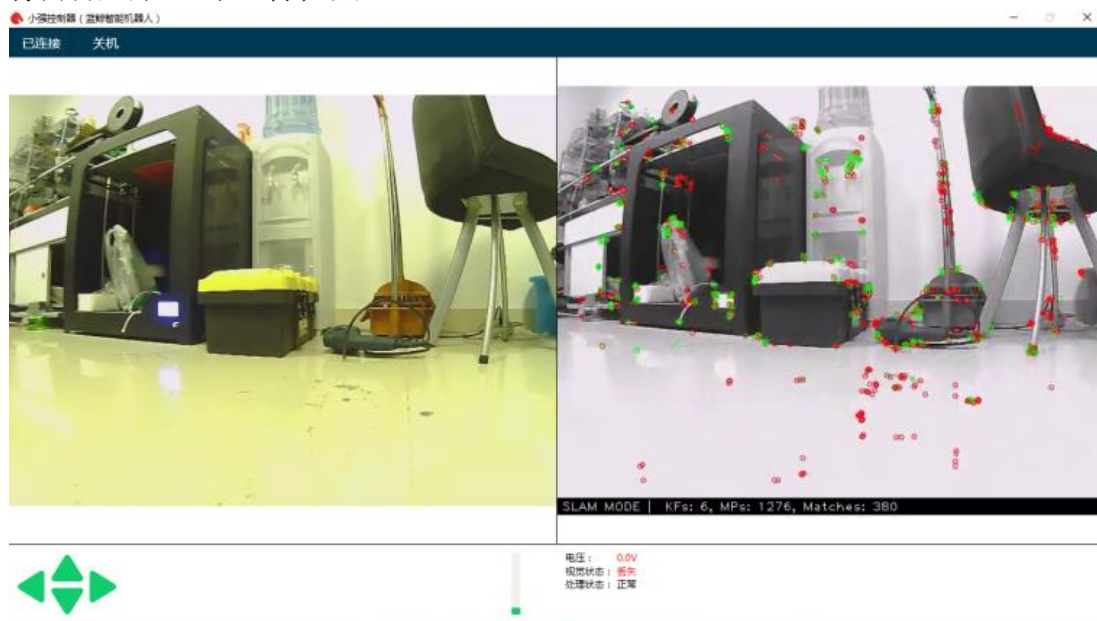


2. 开始建立环境三维模型

打开小强图传遥控 windows 客户端，点击“未连接”按钮连接小强。在图传窗口右键打开“原始图像”和“ORB_SLAM2 的特征点图像”



上图中，左侧图像是摄像头原始彩色图像，右侧是 ORB_SLAM2 处理后的黑白图像。当前 ORB_SLAM2 还没有初始化成功，所以黑白图像没有特征点。按住“w”键开始遥控小强往前缓慢移动，使 ORB_SLAM2 初始化成功，即黑白图像开始出现红绿色特征点



现在就可以开始遥控小强对周围环境建图，遥控过程中需要保证黑白图像一直存在红绿色特征点，不存在则说明视觉 lost 了，需要遥控小强退回到上次没 lost 的地方找回视觉位置。

[客户端操作演示视频](#)

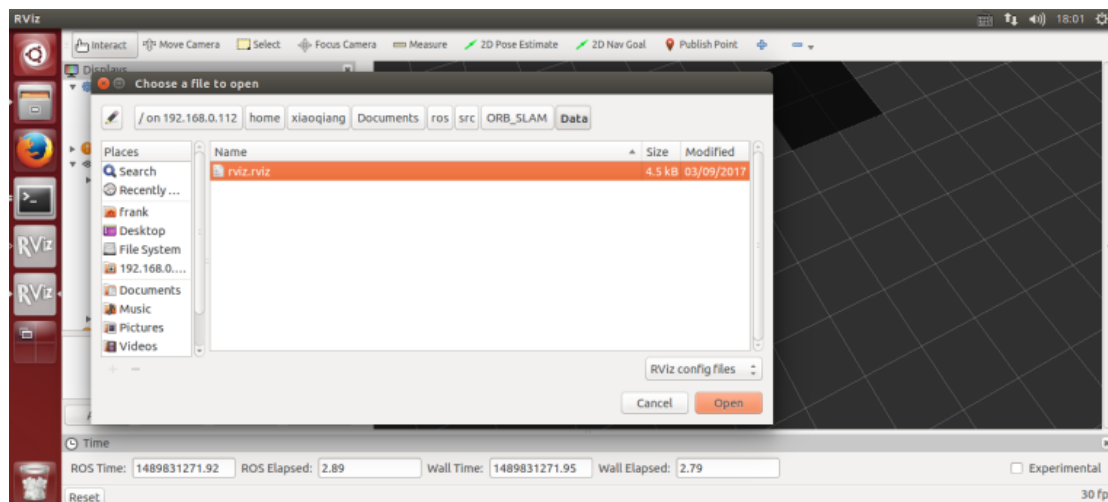
3. 使用 rviz 查看建图效果

在本地虚拟机 hosts 中添加小强 ip 地址,然后新开一个终端打开 rviz

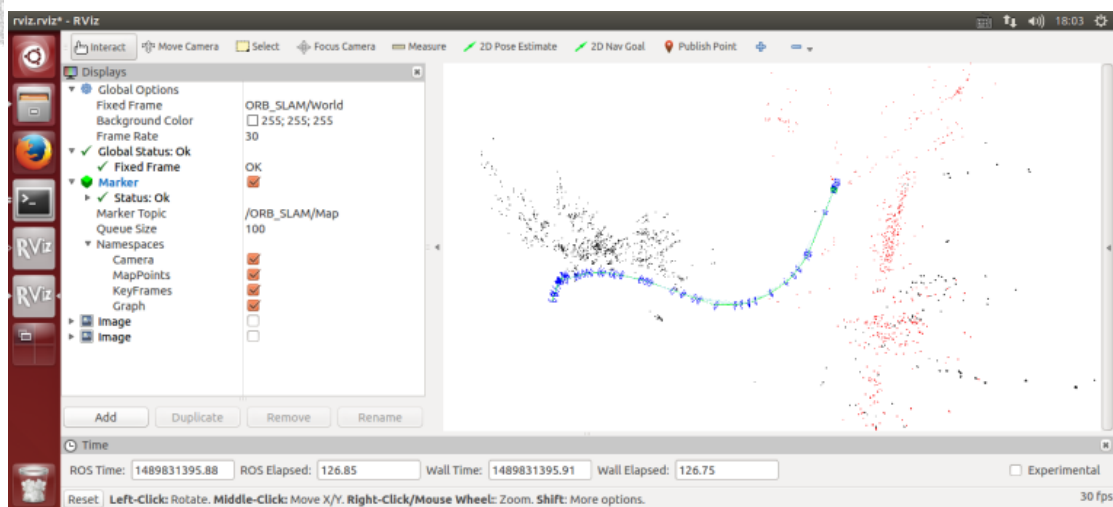
```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
```

rviz

打开/home/xiaoqiang/Documents/ros/src/ORB_SLAM/Data/rviz.rviz 配置文件



如下图所示, 红黑色点是建立的三维模型(稀疏特征点云), 蓝色方框是 keyframe 可以表示小强轨迹



4. 保存地图

当地图建立的范围满足自己要求后, 在虚拟机新开一个命令窗口, 输入下列命令保存地图

```
ssh -X xiaoqiang@192.168.xxx.xxx
```

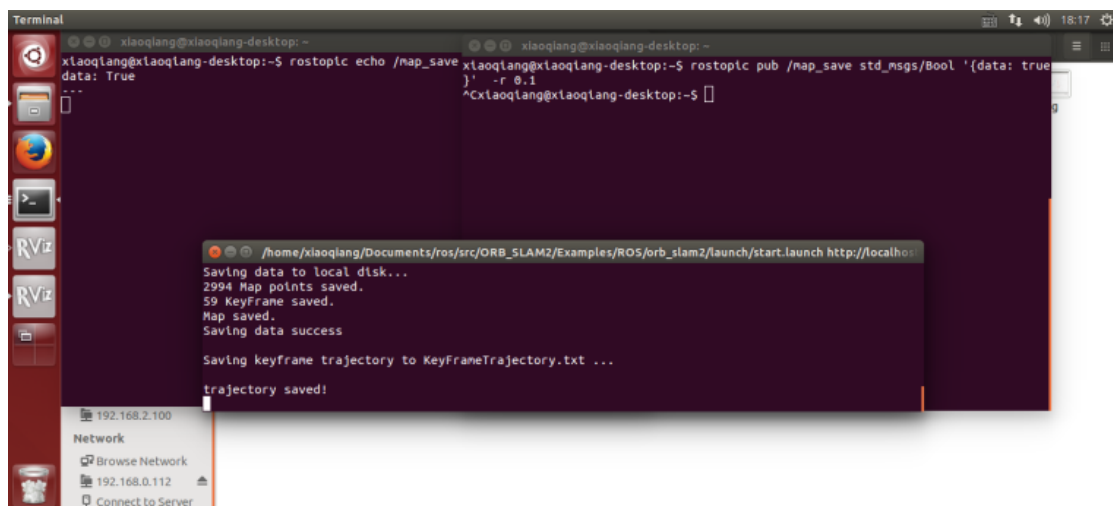
```
rostopic pub /map_save std_msgs/Bool '{data: true}' -r 0.1
```

这个命令每隔 10s 触发 1 次保存任务, 因此当看到下图时, 请关闭上面的窗口

在虚拟机新开一个命令窗口, 输入下列命令侦测地图保存命令有没有发出

```
ssh -X xiaoqiang@192.168.xxx.xxx
```

```
rostopic echo /map_save
```



地图文件会被保存进用户主目录的 `slamdb` 文件夹内。

5. 地图的载入

保存地图之后可以再次载入 ORB_SLAM2 程序中。地图载入后程序可以非常迅速的定位出摄像头的具体位置。这样就可以开发出基于视觉定位的导航算法了。载入地图的方式也非常简单。

将 `/home/xiaoqiang/Documents/ros/workspace/src/ORB_SLAM2/Examples/ROS/orb_slam2/Data` 文件夹内的 `setting.yaml` 文件中的 `LoadMap` 的值设置为 1。

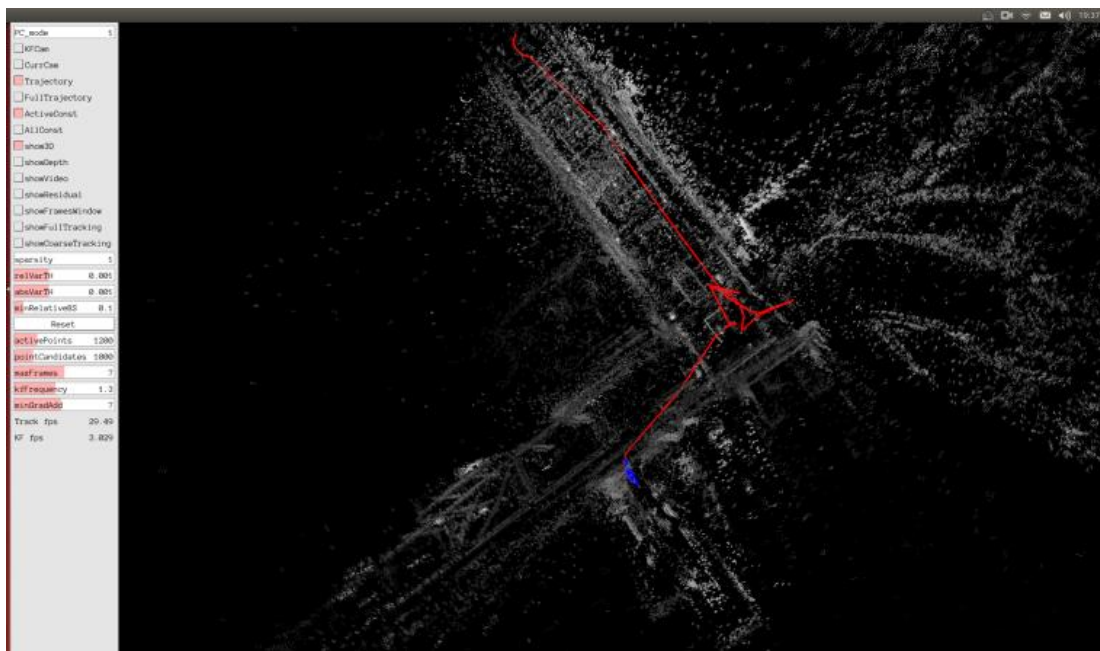
6. 地图的后期处理

在建立地图之后，想要使用这个地图进行导航，还需要对地图文件做进一步的操作。比如创建导航的线路，标记导航路径的行走方式等等。这一部分内容可以参见这篇文章[视觉导航路径编辑器使用教程](#)。

教程(18)___利用 DSO_SLAM 建立环境三维模型

[Direct Sparse Odometry\(DSO\)](#)是业内很流行的 `Isd_slam` 系统作者的学生 Jakob Engel 开发的,实测性能和精度优于 `Isd_slam`。DSO 上个月被作者开源到 [github](#), 同时还一并开源了 DSO 在 `ros` 系统下的使用代码实例 [dso_ros](#)。

本篇教程将演示如何在小强开发平台上安装 DSO 和 `dso_ros`, 利用小强平台上的摄像头实时运行 DSO 进行三维建模, 先上教程的最后测试视频 (效果很不错)。



1. DSO 的安装

注: 因为小强开发平台已经提前安装好了不少 DSO 需要的依赖包, 下文将跳过这些包的安装, 请其它开发平台的读者参考 [github](#) 上的完整安装教程进行安装。

1.A 安装依赖包

```
sudo apt-get install libsuitesparse-dev libeigen3-dev libboost-dev
sudo apt-get install libopencv-dev
```

1.B 下载源代码

```
cd ~/Documents/
git clone https://github.com/JakobEngel/dso.git
```

1.C 继续配置依赖包

```
sudo apt-get install zlib1g-dev
cd ~/Documents/dso/thirdparty
tar -zxvf libzip-1.1.1.tar.gz
```

```
cd libzip-1.1.1/  
./configure  
make  
sudo make install  
sudo cp lib/zipconf.h /usr/local/include/zipconf.h
```

1.D 编译安装

```
cd ~/Documents/dso/  
mkdir build  
cd build  
cmake ..  
make -j
```

2. dso_ros 的安装

注：原作者提供的源代码有两个分支，master 分支对应 rosbuild 版，catkin 分支对应 catkin 版。对于现代 ROS 版本，推荐使用 catkin 版本，安装使用更方便。但是作者的 catkin 分支存在代码缺陷，实际无法安装使用，因此下文将安装我们蓝鲸智能修改之后的 [dso ros 版本](#)。

```
cd ~/Documents/ros/src  
git clone https://github.com/BlueWhaleRobot/dso_ros.git  
cd ..  
export DSO_PATH=/home/xiaoqiang/Documents/dso  
catkin_make
```

3. 开始使用

注：小强开发平台的摄像头标定文件是相同的，因此可以直接运行下列命令，请其它开发平台的读者自行修改 camera.txt 文件中的内容（留意每行的最后结尾不能有空格）以及命令中的 image topic 名字

```
roslaunch dso_ros dso_live image:=/camera_node/image_raw calib=/home/xiaoqiang/Documents/ros/src/dso_ros/camera.txt mode=1
```

现在移动摄像头，就能开始对周围环境进行三维建模，移动过程避免急转弯和剧烈运动。对于小强用户，请先遥控小强运动同时用 rosbag 录制 /camera_node/image_raw 这个 image topic 数据，然后重放，这样可以实现大范围的建模。rosbag 重放前，需要关闭 usb 摄像头节点 (sudo service startup stop)，否则会有图像发布冲突。

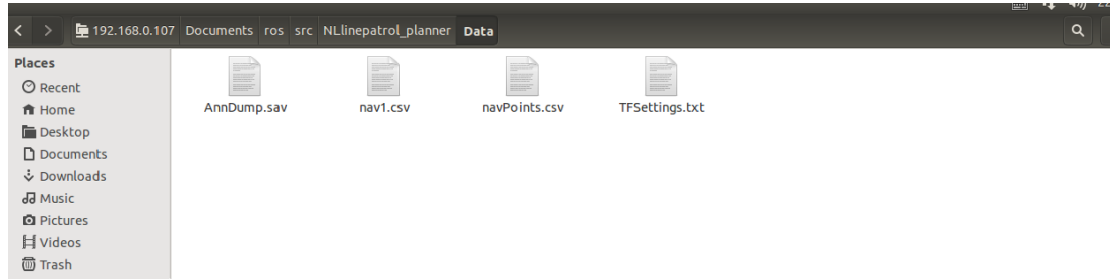
教程(19)___Nllinepatrol_planner 的简单使用

随小车主机附带的 Nllinepatrol_planner 是一个用于视觉导航的全局路径规划器, 根据小车输出的视觉轨迹 (视觉轨迹文件的获得请参考当前栏目下的帖子), 能输出一条连接小车当前坐标和目的坐标的全局路径, 下文将通过一个模拟实例来演示它的使用方式。

主要思路是: 一个 python 脚本发布虚拟的视觉里程计和相关 tf 树, 一个 python 脚本给 move_base 节点发布目标点, 最后 move_base 节点通过调用 Nllinepatrol_planner 获得一条全局路径并在 rviz 中显示。

1. 配置 Nllinepatrol_planner

需要提供 Nllinepatrol_planner 待读取的[视觉导航路径文件]、轨迹坐标变换需要的变换参数文件, 这两文件都应该放在 Nllinepatrol_planner 下面的 data 文件夹内, 文件名任意, 通过配置 move_base 中的相关参数可以指定 Nllinepatrol_planner 读取的文件, 下文会说明。



在上图中, NAV1.CSV 是视觉轨迹文件、TFSETTINGS.TXT 为变换参数文件(第一行是旋转矩阵的 9 个元素、数组元素排列顺序为 C 语言的行排列, 第二行为平移向量的 XYZ 分量, 第三行为 SCALE 因子)

2. 制作 move_base 的 launch 文件

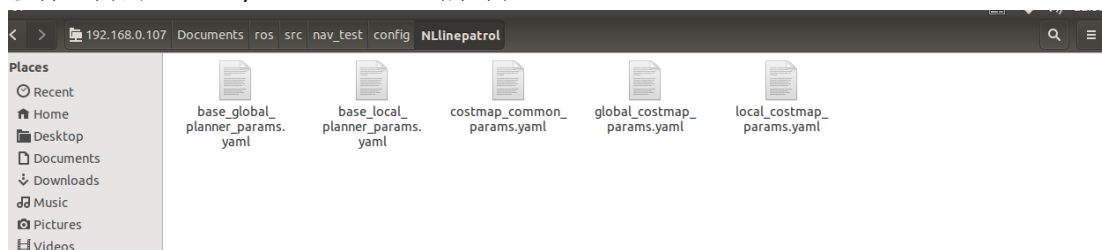
对于本教程, 我们已经在 nav_test 软件包中的 launch 文件夹内提供了相关的 luanch 文件, 名字为 xq_move_base_blank_map2.launch, 这个 luanch 文件在实际运用时可以作为模板, 需要注意的地方请看下图



这个 `launch` 文件会调用 `xq_move_base2.launch` 文件, `xq_move_base2.launch` 文件也在当前目录下, 里面的内容如下

```
<launch>
<node pkg="move_base" type="move_base" respawn="false" name="move_base" output="screen">
  <param name="base_global_planner" value="NLinepatrol_planner/NLinepatrolPlanner"/>
  <roscppparam file="$(find nav_test)/config/NLinepatrol/costmap_common_params.yaml" command="load" ns="global_costmap" />
  <roscppparam file="$(find nav_test)/config/NLinepatrol/costmap_common_params.yaml" command="load" ns="local_costmap" />
  <roscppparam file="$(find nav_test)/config/NLinepatrol/local_costmap_params.yaml" command="load" />
  <roscppparam file="$(find nav_test)/config/NLinepatrol/global_costmap_params.yaml" command="load" />
  <roscppparam file="$(find nav_test)/config/NLinepatrol/base_local_planner_params.yaml" command="load" />
  <roscppparam file="$(find nav_test)/config/NLinepatrol/base_global_planner_params.yaml" command="load" />
</node>
</launch>
```

通过上述内容, 可以发现通过设置 `BASE_GLOBAL_PLANNER` 参数值来制定全局路径规划器为 `NLLINEPATROL_PLANNER`, 还可以看出 `MOVE_BASE` 的其它参数配置文件放在 `NAV_TEST` 软件包内的 `CONFIG/NLLINEPATROL` 路径内



对于上图, 我们需要更改的文件是 `BASE_GLOBAL_PLANNER_PARAMS.YAML`, 因为这个文件里的内容对应 `NLLINEPATROL_PLANNER` 运行时实际加载的参数, 默认内容如下

`NLinepatrolPlanner:`

```
DumpFileName: AnnDump.sav
strTFParsFile: TFSettings.txt
TxtFileName: nav1.csv
ANN_Dump_Bool: false
connect_distance: 0.3
```

`TXTFILENAME` 指定加载的视觉轨迹文件名, `STRTFPARSFILE` 指定加载的变换参数文件名, `CONNECT_DISTANCE` 设置视觉轨迹文件中连通点之间的最大距离 (坐标变换后两个点之间的距离小于该值就认为这两点之间没有障碍物, 可以直接连接), `ANN_DUMP_BOOL` 值为 `FALSE` 表示从 `TXT` 文件中加载轨迹和变换参数、如果为 `TRUE` 则从 `DUMPFILNAME` 参数指定的 `DUMP` 文件加载 (多次使用同一个视觉轨迹文件时, 第二次以后从 `DUMP` 文件启动可以加速)

3. 配置完成开始使用

A. 因为我们这次是虚拟运行, 发布的一些 `topic` 是没有实际意义的但是和小强默认 `ROS` 驱动冲突, 所以现在需要停止所有 `ROS` 运行实例

```
sudo service startup stop
roscore
```

B. 启动虚拟 topic 和小强模型文件

```
roslaunch orb_init temp.py //发布 odom
```

```
roslaunch xiaoqiang_udrf xiaoqiang_udrf.launch //启动模型
```

C. 启动上文制作的 xq_move_base_blank_map2.launch 文件

```
roslaunch nav_test xq_move_base_blank_map2.launch
```

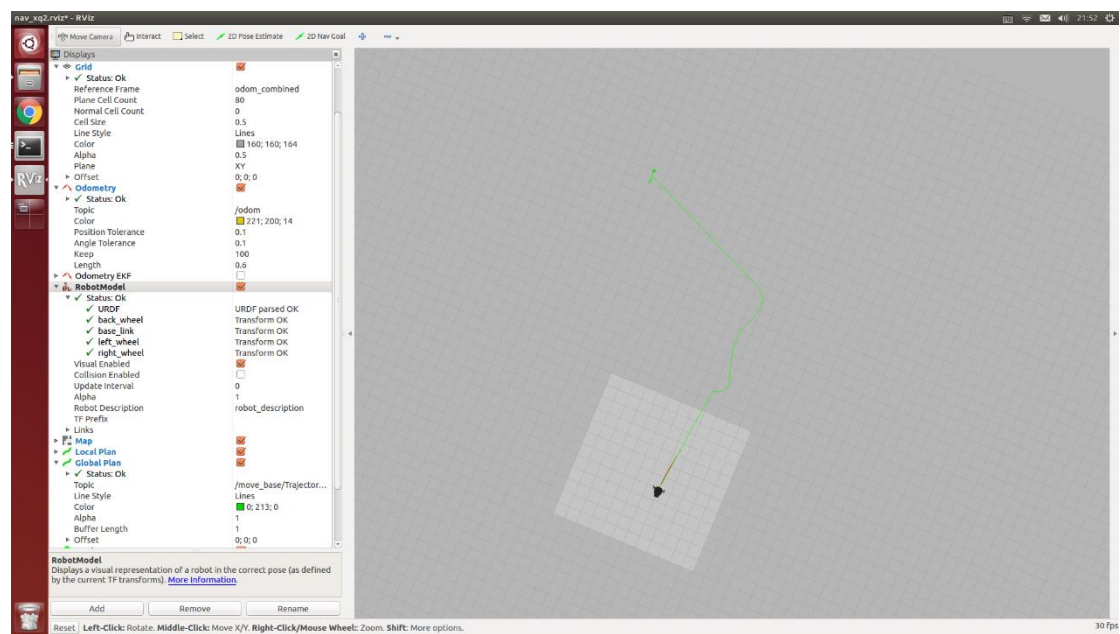
D. 启动 rviz，并打开 ros/src/nav_test/config/nav_xq2.rviz 配置文件

Rviz

E. 启动虚拟 goal 发布节点 (基于惯性导航中的 squre.py 修改而来)

```
roslaunch nav_test NLinepatrol.py
```

4. 现在 rviz 中就已经出现目标全局路径轨迹了(绿线)，想测试其它 goal 目标点，请自行修改 NLinepatrol.py 中相关代码



教程(20) 获取小车视觉里程计并在 rviz 中显示小车轨迹

1. 本机 ssh 准备部分 (小强主机是受控端, 本机是指远程控制端)

1.1 ssh 远程登录小强主机, 下文的操作如果不加特殊声明则都是在这个 ssh 窗口中输入的

```
ssh xiaoqiang@xxx.xxx.xxx.xxx
```

请将 xxx.xxx.xxx.xxx 换成小强当前实际的 ip 地址

1.2 查看 startup 开机任务是否运行

```
sudo service startup status
```

如果显示 runing, 说明正常, 如果显示 stopped, 则重新启动它

```
sudo service startup start
```

如果想关闭这个任务, 可以使用这条指令

```
sudo service startup stop
```

1.3 查看系统状态

```
rostopic echo /system_monitor/report
```

如果正常, 则显示如下

```
imageStatus: TrueodomStatus: TrueorbStartStatus: FalseorbInitStatus: FalseorbScaleStatus:
Falsebrightness: 0power: 12.34432
```

1.4 如果不正常请重启 startup 开机任务

```
sudo service startup restart
```

1.5 新开一个命令窗口启动 ORB_SLAM

```
ssh xiaoqiang@xxx.xxx.xxx.xxx
```

```
roslaunch ORB_SLAM ov2610.launch
```

1.6 回到上一个窗口输入下述指令, 等待 ORB_SLAM 启动完毕

```
rostopic echo /system_monitor/report
```

如果 ORB_SLAM 启动完毕则有如下内容显示:

```
orbStartStatus: True
```

2. 本机本地操作部分

本机已安装机器人系统 ros jade 版本，电脑操作系统是 ubuntu14.04，ros 的安装可以参考这篇教程的前半部分

2.1 将本机加入小强的 ros 局域网

在本地开启一个命令行终端，在本地/etc/hosts 文件内添加小强的 ip

```
sudo gedit /etc/hosts
```

添加

```
xxx.xxx.xxx.xxx xiaoqiang-desktop
```

保存退出

请将 xxx.xxx.xxx.xxx 换成小强当前实际的 ip 地址

2.2 加入 ros 局域网

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
```

```
rostopic list
```

如果加入成功，命令行会输出小强主机上的 topic，更多关于多台 ros 机器联机的设置教程请[移步这里](#)

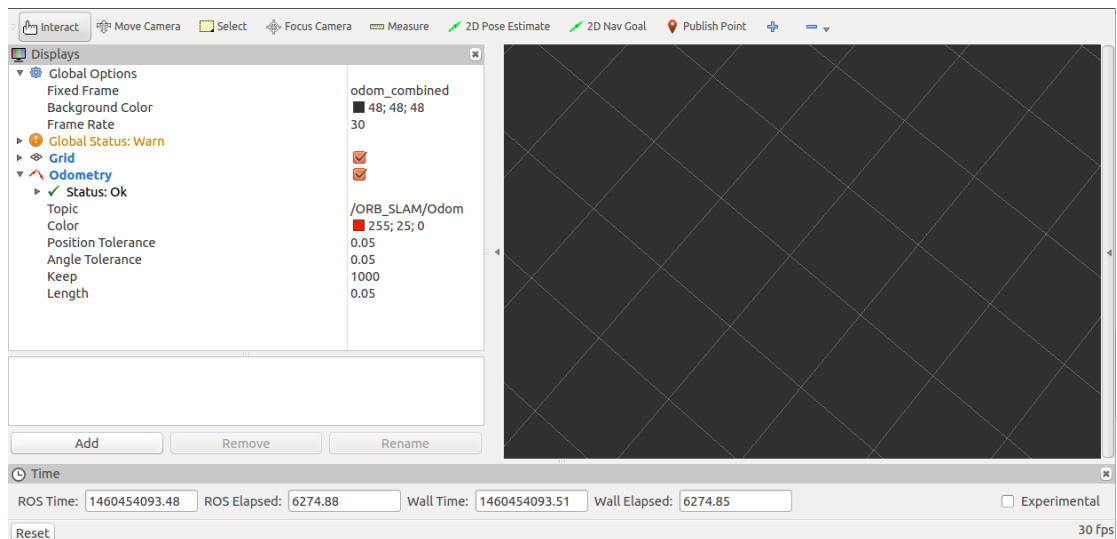
2.3 打开 rviz

先下载 rviz [配置文件](#)，这个配置文件也可直接从小强主机上的 orb_init 包中拷贝(推荐)，用于查看小强视觉系统输出的轨迹。

在本地命令行终端中输入

```
rviz
```

当窗口打开后，点击左上角的 file->open，选择上述下载的配置文​​件。这时界面应该如下图所示



3. 在新开一个 ssh 窗口，用于启动小强主机上的 orb_init

启动前确保小强周围有两平米大小的自由空间，小强会自主移动一段时间

```
roslaunch orb_init orb_scale.py
```

orb_init 初始化完成后不能关闭 orb_init，它会持续输出视觉里程计 topic，这个 topic 就是本机 rviz 需要显示的内容。

新开一个窗口查看系统状态

```
rostopic echo /system_monitor/report
```

如果正常，则显示如下

```
imageStatus: TrueodomStatus: TrueorbStartStatus: TrueorbInitStatus: TrueorbScaleStatus: TrueorbBrightness: 0power: 12.34432
```

到此我们就已经获取小车的视觉里程计

4. 新开一个 ssh 窗口，用于控制小车移动

```
roslaunch nav_test control.py
```

通过方向键来控制小强的移动。空格键是停止。Ctrl + C 退出程序。

随着小强的移动，本机上的 rviz 界面里就会实时更新显示出小车的轨迹,我们自己的测试视频[\[请点击观看\]](#)

如果 rviz 无变化，在 ssh 中 screen 下新开一个窗口，文本输出小车轨迹。

```
rostopic echo /ORB_SLAM/Camera
```

教程(21)___获取 usb 摄像头 30fps 的 1080p 图像流及 120fps 的 VGA 分辨率图像流

小车默认开机启动的 usb 摄像头节点，发布的是 30fps 的 VGA (640*480) 分辨率的图像流，可以满足大部分的视觉任务要求。usb 摄像头硬件实际可以输出 120fps 的 VGA 图像流和高达 30fps 的 1080p 图像流，本文将介绍获取这两种规格图像的办法。

1. 升级 usb_cam 节点源代码

由于 usb2.0 带宽问题，输出 120fps 的 VGA 视频和 30fps 的 1080p 视频，需要采用 mjpeg 格式。小车自带的 usb 摄像头 ROS 驱动包 usb_cam，它不能以 mjpeg 方式读取小车 usb 摄像头，因此我们需要先升级 usb_cam。升级办法请参考[这篇帖子](http://community.bwbot.org/topic/144/解决 ROS 的 usb_cam 节点无法正常读取 mjpeg 格式摄像头的方法):http://community.bwbot.org/topic/144/解决 ROS 的 usb_cam 节点无法正常读取 mjpeg 格式摄像头的方法

2. 测试 120fps 的 VGA 视频输出

首先关闭小车的开机启动任务

```
sudo service startup stop
```

修改 usb_cam 包中的 ov2610mjpg.launch 文件内容为下文所示

```
<launch>
  <node name="camera_node" pkg="usb_cam" type="usb_cam_node">
    <param name="video_device" value="/dev/video0" />
    <param name="image_width" value="640" />
    <param name="image_height" value="480" />
    <param name="framerate" value="120" />
    <param name="pixel_format" value="mjpeg" />
    <param name="camera_frame_id" value="ov2610" />
    <param name="io_method" value="mmap"/>
  </node>
</launch>
```

启动上述 launch 文件

```
roslaunch usb_cam ov2610mjpg.launch
```

新开一个窗口打印发布的图像 topic 帧率

```
rostopic hz /camera_node/image_raw
```

正常会出现类似下图的输出

```
subscribed to [/camera_node/image_raw]
average rate: 99.497
  min: 0.004s max: 0.016s std dev: 0.00246s window: 98
average rate: 99.324
  min: 0.004s max: 0.016s std dev: 0.00240s window: 198
average rate: 99.385
  min: 0.004s max: 0.016s std dev: 0.00236s window: 297
average rate: 99.247
  min: 0.004s max: 0.016s std dev: 0.00236s window: 396
average rate: 99.302
  min: 0.004s max: 0.016s std dev: 0.00232s window: 495
average rate: 99.296
  min: 0.004s max: 0.016s std dev: 0.00231s window: 595
average rate: 99.249
  min: 0.004s max: 0.016s std dev: 0.00230s window: 694
```

上文输出显示摄像头帧率大概是 99fps，并没有达到 120fps，这是因为环境光照的影响，如果环境光照亮度充足时是可以达到 120 帧的。

2. 测试 30fps 的 1080p 视频输出

首先关闭小车的开机启动任务

```
sudo service startup stop
```

修改 usb_cam 包中的 ov2610mjpg.launch 文件内容为下文所示

```
<launch>
  <node name="camera_node" pkg="usb_cam" type="usb_cam_node">
    <param name="video_device" value="/dev/video0" />
    <param name="image_width" value="1920" />
    <param name="image_height" value="1080" />
    <param name="framerate" value="30" />
    <param name="pixel_format" value="mjpeg" />
    <param name="camera_frame_id" value="ov2610" />
    <param name="io_method" value="mmap"/>
  </node>
</launch>
```

启动上述 launch 文件

```
roslaunch usb_cam ov2610mjpg.launch
```

新开一个窗口打印发布的图像 topic 帧率

```
rostopic hz /camera_node/image_raw
```

正常会出现类似下图的输出

```
xiaoqiang@xiaoqiang-desktop:~$ rostopic hz /camera_node/image_raw
subscribed to [/camera_node/image_raw]
average rate: 29.986
  min: 0.028s max: 0.039s std dev: 0.00318s window: 30
average rate: 29.917
  min: 0.028s max: 0.039s std dev: 0.00296s window: 59
average rate: 29.912
  min: 0.028s max: 0.039s std dev: 0.00292s window: 89
average rate: 29.853
  min: 0.027s max: 0.042s std dev: 0.00308s window: 119
average rate: 29.874
  min: 0.027s max: 0.042s std dev: 0.00283s window: 149
average rate: 29.839
  min: 0.027s max: 0.042s std dev: 0.00282s window: 179
^Caverage rate: 29.849
  min: 0.027s max: 0.042s std dev: 0.00282s window: 202
```


教程(22) 操作 6 自由度机械臂

6 自由度机械臂的资料在这个百度云盘[链接](http://pan.baidu.com/s/1nuBpDaD)里：<http://pan.baidu.com/s/1nuBpDaD> 请仔细阅读二次开发部分的串口通信协议、虽然下文使用的是 `usb hid` 通信方式，但命令格式是一样的。下文将操作 6 自由度机械臂完成主控板上存储的 3 组动作。

控制原理：小车主机使用 `usb` 连接机械臂主控板，用户发送名为 `robot_arm/cmdstring` 的 `topic`，这个 `topic` 内容为控制命令，最后由 `robot_arm` 节点负责将这个 `topic` 内容经由 `usb hid` 协议发送给机械臂主控板。

1. 运行 `robot_arm` 节点

```
roslaunch robot_arm move.py
```

启动正常，机械臂会执行动作 0（默认状态），同时命令窗口会显示如下内容

```
xiaoqiang@xiaoqiang-desktop:~/Documents/ros$ roslaunch robot_arm move.py
```

```
Opening robot arm deviceManufacturer: MyUSB_HIDProduct: LOBOT
```

```
Serial No: 8D9823654852
```

```
Run the zero group action
```

2. 构造 `robot_arm/cmdstring` topic

`robot_arm/cmdstring` 这个 `topic` 类型是 `std_msgs/msg` 中的 `String` 即字符串。根据上文提供的机械臂二次开发资料，我们可以知道 6 自由度的控制协议是由一个无符号 `byte` 数组表示，因此我们在这里稍加改造，直接将这个数组的 `hex` 值变成字符串打包成 `topic` 命令。变换方式是采用 `python` 的 `byte array` 表示方式，将数组中每个元素的 `hex` 编码串在一起，然后将其中的 `0x` 换成 `\x`。

例如：`[0x55,0x55,0x05,0x06,0x00,0x01,0x00]`，这个控制命令数组

转换成 `robot_arm/cmdstring` 的内容是 `'\x55\x55\x05\x06\x00\x01\x00'`。

小技巧：看不懂的话，还可以用 `python` 的 `map` 函数帮忙转换，找规律：
`map(ord, '\x55\x55\x05\x06\x00\x01\x00')`

我们要控制机械臂完成 3 个动作，这三个字符串内容如下

```
'\x55\x55\x05\x06\x00\x01\x00'
```

```
'\x55\x55\x05\x06\x01\x01\x00'
```

```
'\x55\x55\x05\x06\x02\x01\x00'
```

3. 发布运动命令

新开一个命令行终端，因为是演示，所以直接使用 rostopic 的 pub 功能，将上面的字符串命令打包成 topic 发给 robot_arm 节点

动作 1: rostopic pub robot_arm/cmdstring std_msgs/String '\x55\x55\x05\x06\x00\x01\x00' -r 0.1

动作 2: rostopic pub robot_arm/cmdstring std_msgs/String '\x55\x55\x05\x06\x01\x01\x00' -r 0.1

动作 3: rostopic pub robot_arm/cmdstring std_msgs/String '\x55\x55\x05\x06\x02\x01\x00' -r 0.1

4. 运动结果暂时没有视频，需要自己实际操作，觉得运动不满意的话请参考我们这个舵机运动控制帖子：[电机控制与缓动函数](#)

教程(23)___ROS 入门手册

[Learning ROS for Robotics Programming - Second Edition.pdf](#)(如果点击无法下载请留言邮箱或者浏览器输入 <http://pan.baidu.com/s/1ge6ffZt>)这本教程很基础，虽然以 Hydro 版本为例，但是也完全兼容 jade 版本，代码实例中只需将书中的 Hydro 字符串替换成 jade 即可。

在完成小强 ROS 机器人教程(1)后，如果您不熟悉 ROS，请重点阅读本书的第二章和第三章。

Preface	ix
Chapter 1: Getting Started with ROS Hydro	1
PC installation	4
Installing ROS Hydro – using repositories	4
Configuring your Ubuntu repositories	5
Setting up your source.list file	6
Setting up your keys	7
Installing ROS	7
Initializing rosdep	8
Setting up the environment	9
Getting rosinstall	10
How to install VirtualBox and Ubuntu	11
Downloading VirtualBox	11
Creating the virtual machine	12
Installing ROS Hydro in BeagleBone Black (BBB)	15
Prerequisites	16
Setting up the local machine and source.list file	18
Setting up your keys	19
Installing the ROS packages	19
Initializing rosdep for ROS	20
Setting up the environment in BeagleBone Black	20
Getting rosinstall for BeagleBone Black	21
Summary	21

三、维护

1 充电

将电池与车底盘的连接断开后，用配送的电池专用充电器充电，充满电需要 5 个小时左右，充满后指示灯会由红色变青色。

电池的两根输出线内部是并联在一起的，因此两根头都可以用于充电和放电（充电器只有一个公头），电池支持同时充放电。

2 车轮松动打滑



解决办法：请重新拧紧上图中的螺丝

3 小强底盘固件的自动更新升级方法

2017 年 3 月 15 日之前购买的用户, 请先完成[[这篇帖子](#)](小强底层固件下载和升级办法)的操作.

3.1 升级固件包软件

```
ssh xiaoqiang@192.168.xxx.xxx
cd ~/Documents
ls
# 如果可以看到 stm32loader 文件夹, 则升级
cd stm32loader
git stash
git pull
# 如果不能看到 stm32loader 文件夹, 则重新下载
git clone https://github.com/BlueWhaleRobot/stm32loader.git
cd stm32loader
```

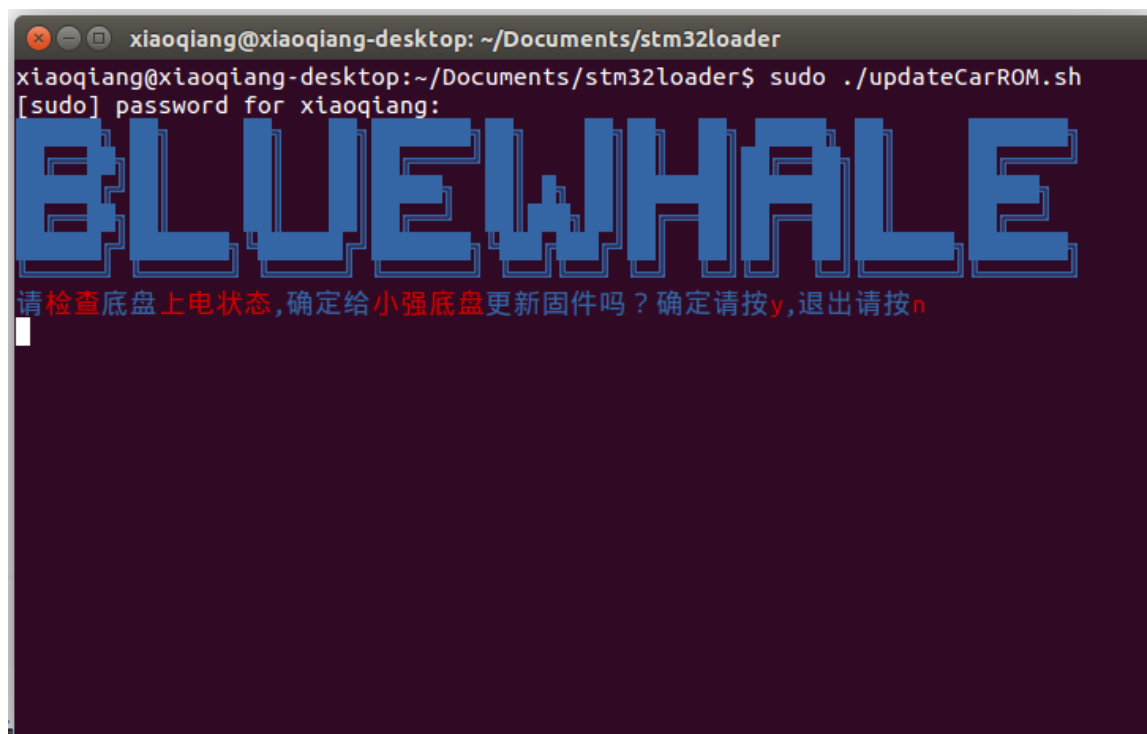
3.2 开始升级底盘固件

对于 2017 年 3 月 11 日之前购买的用户: 由于硬件版本限制, 升级固件过程中可能会导致底盘乱动, 因此请先拔下绿色电机驱动板上中间的“VMOT GND”接线, 升级之后再插回去。

对于 2017 年 3 月 11 日之后购买的用户: 不用插拔任何硬件

```
sudo ./updateCarROM.sh
```

正常的话会出现下图



输入 y,按回车开始更新

```
xiaoqiang@xiaoqiang-desktop: ~/Documents/stm32loader
xiaoqiang@xiaoqiang-desktop:~/Documents/stm32loader$ sudo ./updateCarROM.sh
[sudo] password for xiaoqiang:
BLUEWHALE
请检查底盘上电状态,确定给小强底盘更新固件吗? 确定请按y,退出请按n
y
  停止运行ros任务,释放串口资源
stop: Unknown instance:
  开始更新小强底盘固件...
█
```

```
xiaoqiang@xiaoqiang-desktop: ~/Documents/stm32loader
请检查底盘上电状态,确定给小强底盘更新固件吗? 确定请按y,退出请按n
y
  停止运行ros任务,释放串口资源
stop: Unknown instance:
  开始更新小强底盘固件...
Bootloader version 22
Chip id: 0x410 (xiaoqiang driver_boad_mini)
found xiaoqiang driver_boad_mini
Write 256 bytes at 0x8000000
Write 256 bytes at 0x8000100
Write 256 bytes at 0x8000200
Write 256 bytes at 0x8000300
Write 256 bytes at 0x8000400
Write 256 bytes at 0x8000500
Write 256 bytes at 0x8000600
Write 256 bytes at 0x8000700
Write 256 bytes at 0x8000800
Write 256 bytes at 0x8000900
Write 256 bytes at 0x8000A00
Write 256 bytes at 0x8000B00
Write 256 bytes at 0x8000C00
Write 256 bytes at 0x8000D00
Write 256 bytes at 0x8000E00
█
```

更新成功会出现下图

```
xiaoqiang@xiaoqiang-desktop: ~/Documents/stm32loader
Read 256 bytes at 0x800F300
Read 256 bytes at 0x800F400
Read 256 bytes at 0x800F500
Read 256 bytes at 0x800F600
Read 256 bytes at 0x800F700
Read 256 bytes at 0x800F800
Read 256 bytes at 0x800F900
Read 256 bytes at 0x800FA00
Read 256 bytes at 0x800FB00
Read 256 bytes at 0x800FC00
Read 256 bytes at 0x800FD00
Read 256 bytes at 0x800FE00
Read 256 bytes at 0x800FF00
Read 256 bytes at 0x8010000
Read 256 bytes at 0x8010100
Read 256 bytes at 0x8010200
Read 256 bytes at 0x8010300
Read 256 bytes at 0x8010400
Read 256 bytes at 0x8010500
Read 256 bytes at 0x8010600
Read 256 bytes at 0x8010700
Verification OK
固件升级成功，请重启主机和给底盘重新上电
xiaoqiang@xiaoqiang-desktop:~/Documents/stm32loader$
```

请根据[升级底盘 ROS 驱动包 XQSERIAL_SERVER](#)完成上位机更新操作。最后根据[这篇教程](#)重新校准底盘 IMU。

3.3 升级失败处理办法

如果已经破坏底盘固件，请咨询客服后参考这篇帖子[小强底层固件下载和升级办法](#)进行手动升级。

如果没有破坏底盘固件，请检查串口接线是否牢固，然后给底盘重新上电后再次尝试

4 升级底盘 ros 驱动包 xqserial_server

4.1 ssh 登录小强主机，进入小强 ros 工作目录

```
ssh xiaoqiang@192.168.xxx.xxx #请将 xxx.xxx 换成实际 ip  
cd Documents/ros/src/
```

4.2 进入 ros 驱动包 xqserial_server,更新软件

```
cd xqserial_server/  
git stash  
git pull  
cd ..  
cd ..  
catkin_make
```

4.3 重启 ros 节点，更新完成

```
sudo service startup stop  
sudo service startup start
```

5 重新校准小车底盘 IMU

适用情况:

发货前, 每台小车的底盘 IMU 都已经校准好, 理论上正常使用是不用重新校准地。如果小车经过长期使用后, 发现底盘输出的 odom 角度开始存在严重飘逸现象, 请按下述步骤重新校准底盘 IMU

操作步骤:

5.1 在本地虚拟机中 SSH 登录小车, 输入下述命令

```
ssh xiaoqiang@192.168.0.xxx -X
rostopic echo /imu_cal
```

5.2 在本地虚拟机中新开一个窗口再次 SSH 登录小车, 输入下述命令

```
ssh xiaoqiang@192.168.0.xxx -X
rostopic pub /imu_cal std_msgs/Bool '{data: true}' -r 0.1
```

5.3 等待 10S, 当步骤 1 中地窗口出现下图时, 说明小车已经启动校准程序, 终止步骤 2 中的 TOPIC 发布命令

```
Last login: Tue Oct 25 21:18:11 2016 from 192.168.0.115
xiaoqiang@xiaoqiang-desktop:~$ rostopic echo /imu_cal
data: True
---
```

5.4 等待 2 分钟, IMU 重新标定完成, 现在无需重启小车即可继续正常使用

6 小强系统镜像

里面包含了 jade 版本的 ros 以及各种针对小强的配置和优化。如果小强主机系统被损坏可以尝试用此镜像还原系统，请根据自己小强型号选择对应镜像。

[小强 pro 镜像](#)

生成于 2017-3-3

小强 mini 镜像(暂无)

6.1 镜像使用方法

下面以在虚拟机中如何安装此镜像为例说明镜像的使用方法，在小强主机上推荐先用虚拟机操作一遍

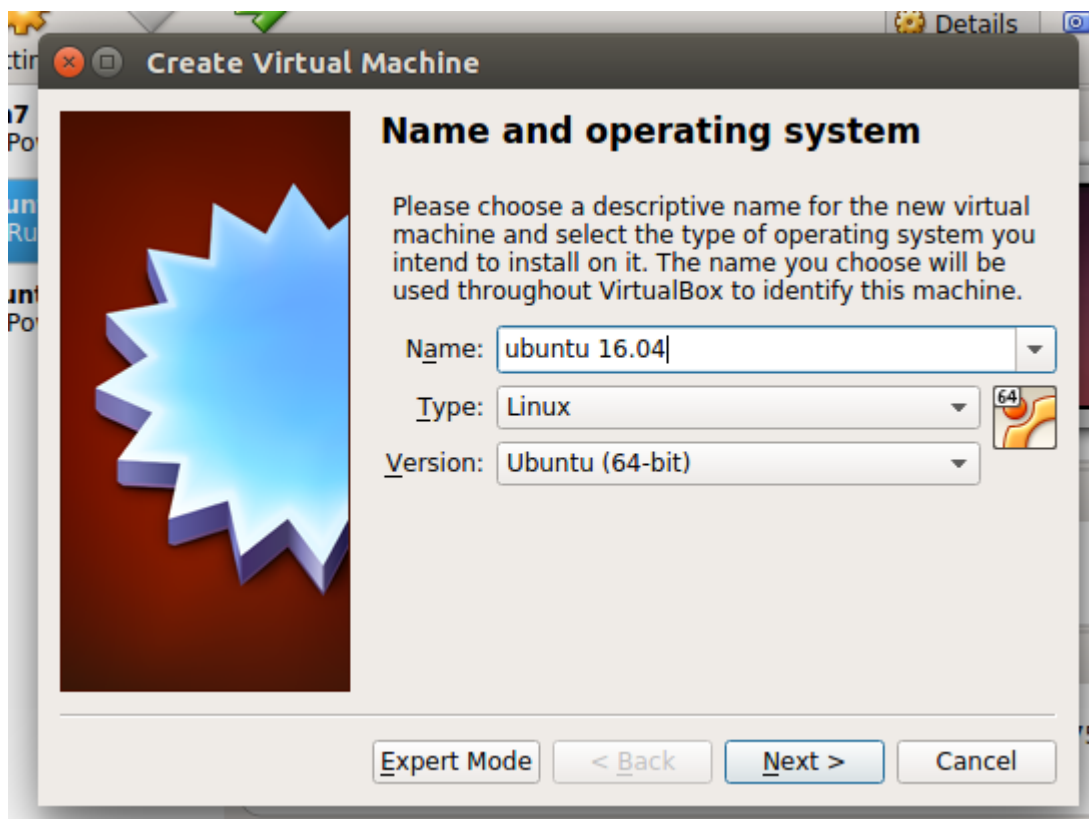
在虚拟机中安装小强系统镜像后，请关闭开机启动项，避免与小强冲突。

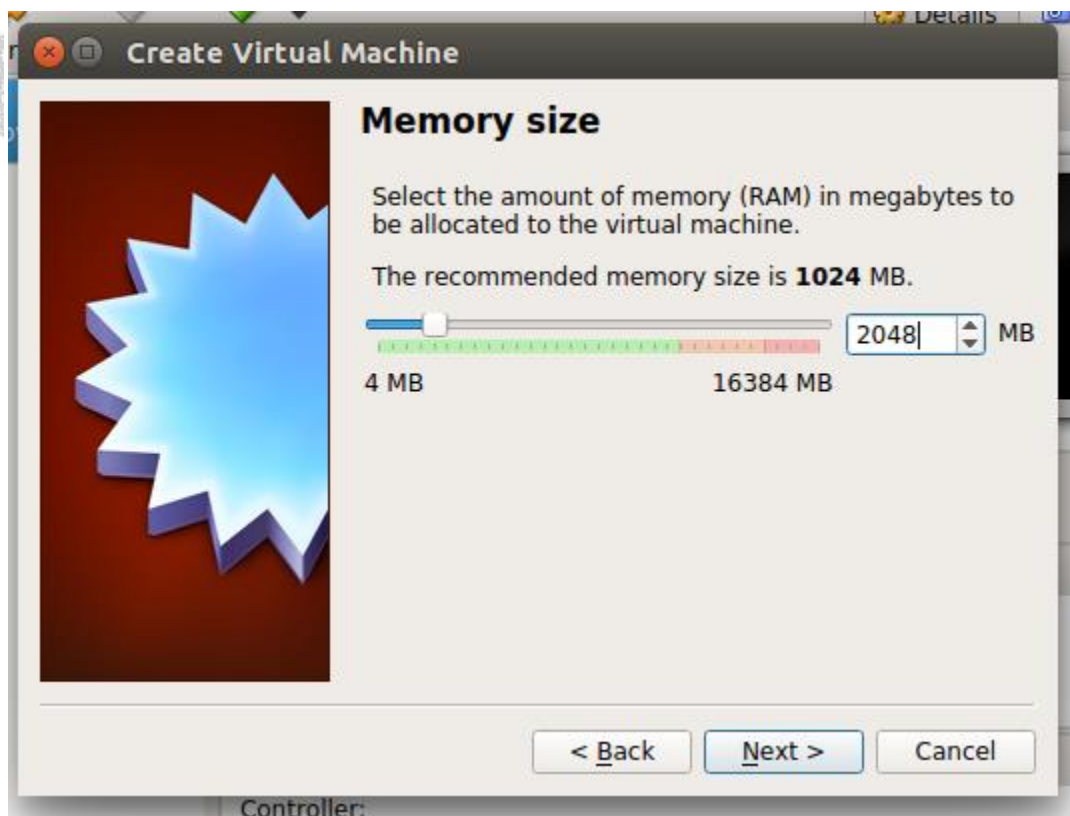
```
sudo service startup stop  
rosrun robot_upstart uninstall startup
```

下载镜像

从以上的链接中下载小强镜像。下载完成之后别忘了进行 md5 校验

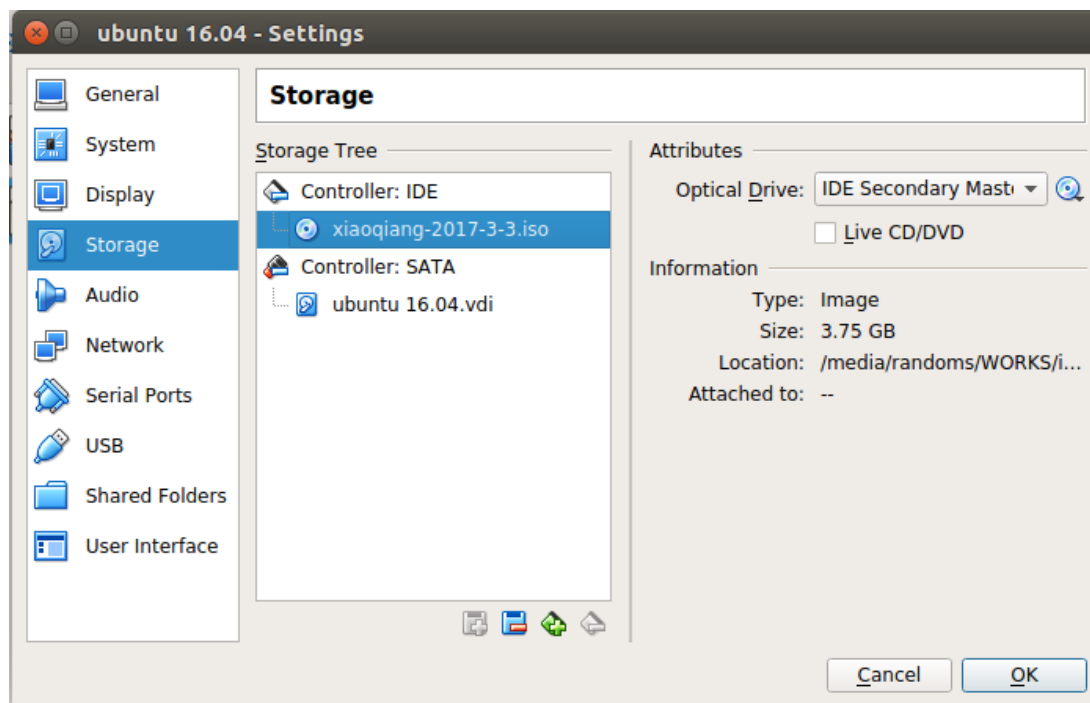
创建虚拟机





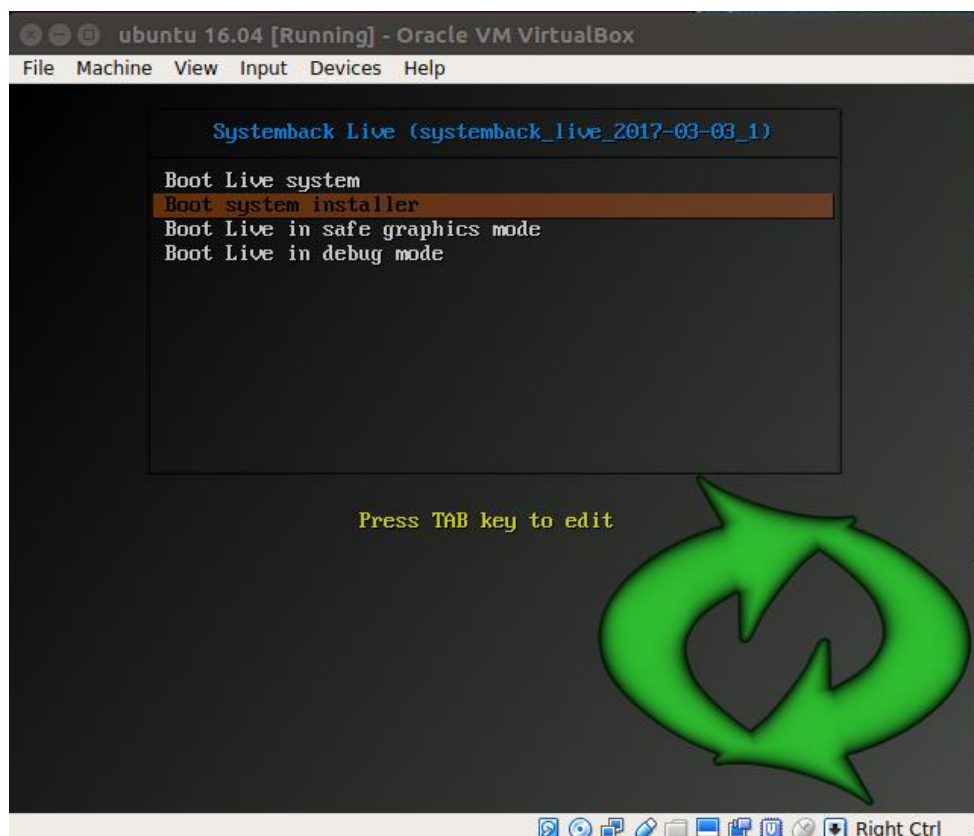
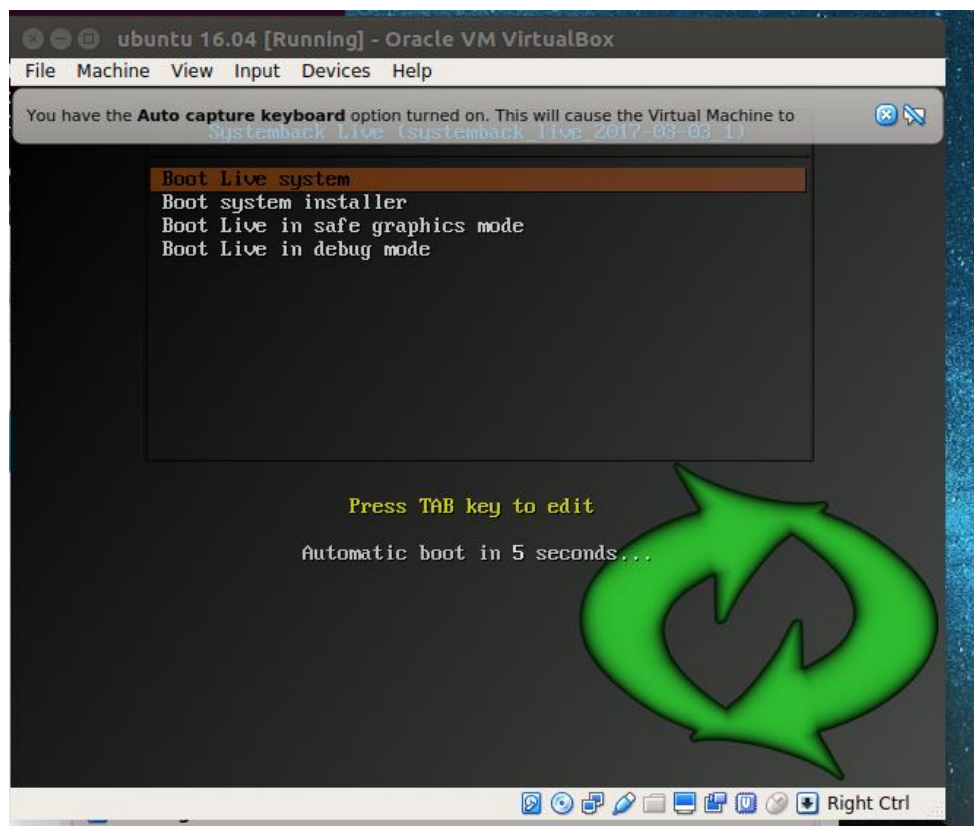


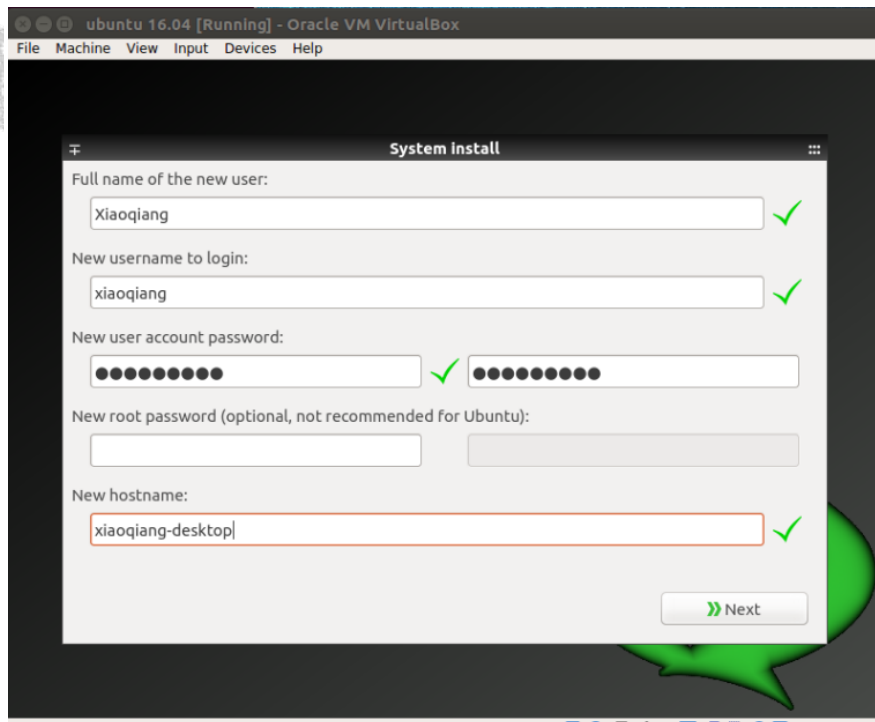
如图所示操作依次选择。给虚拟机添加 iso 文件



开始安装

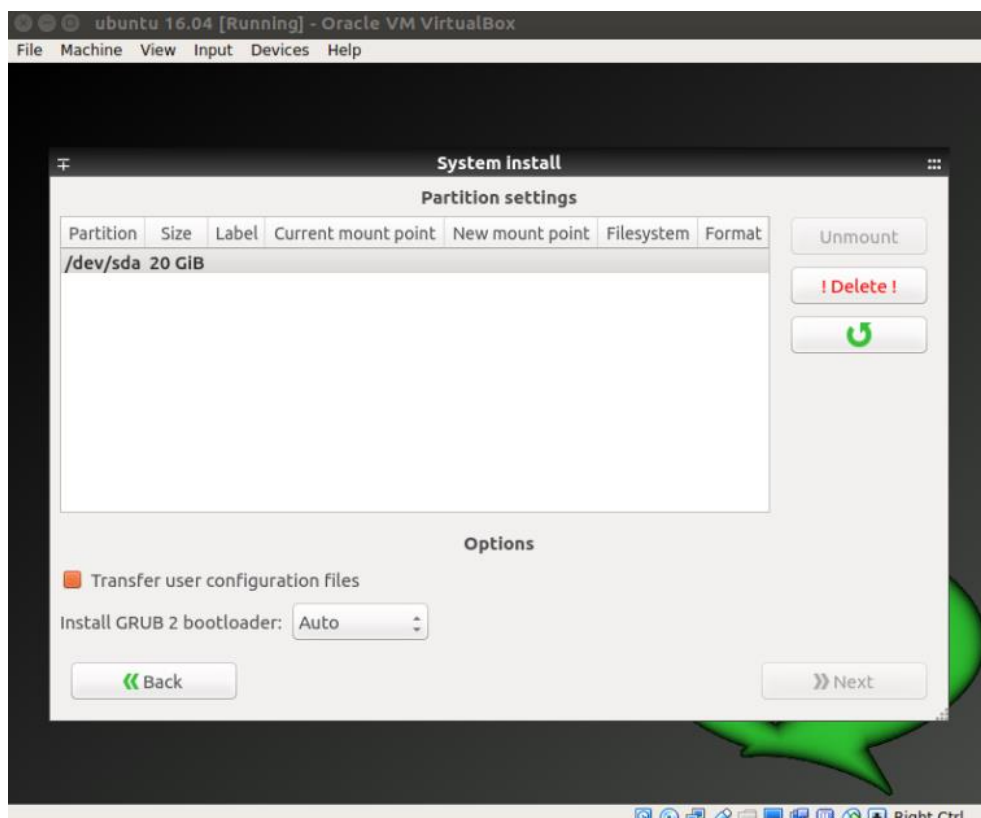
启动虚拟机

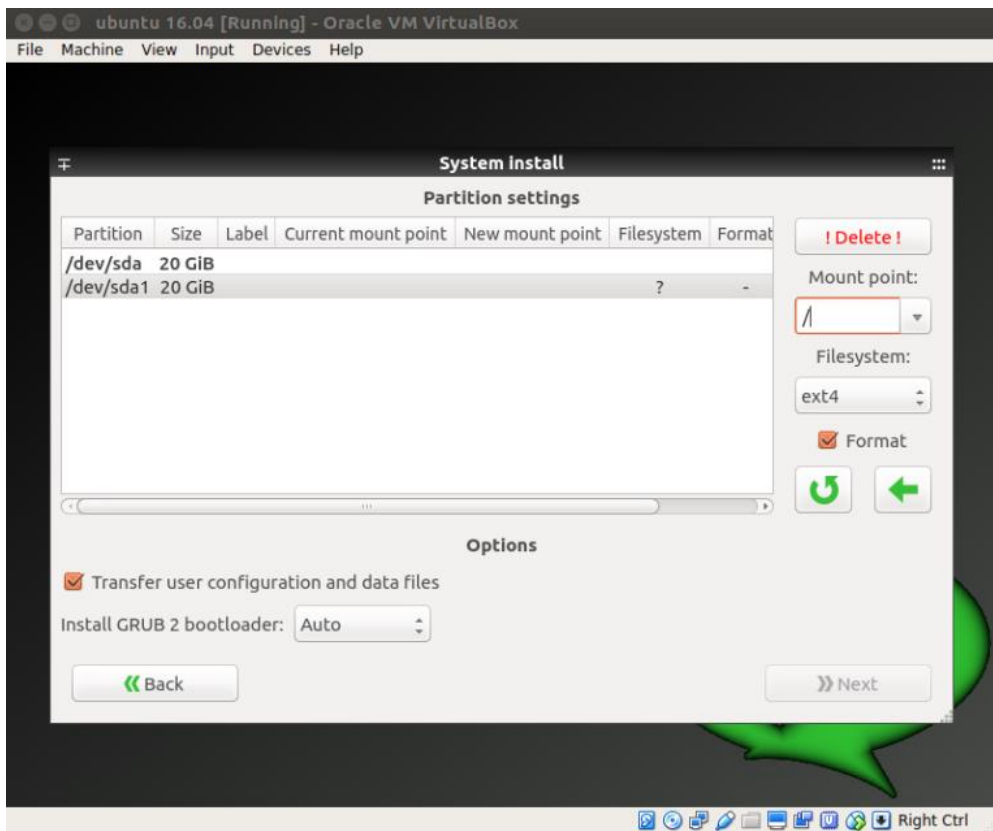
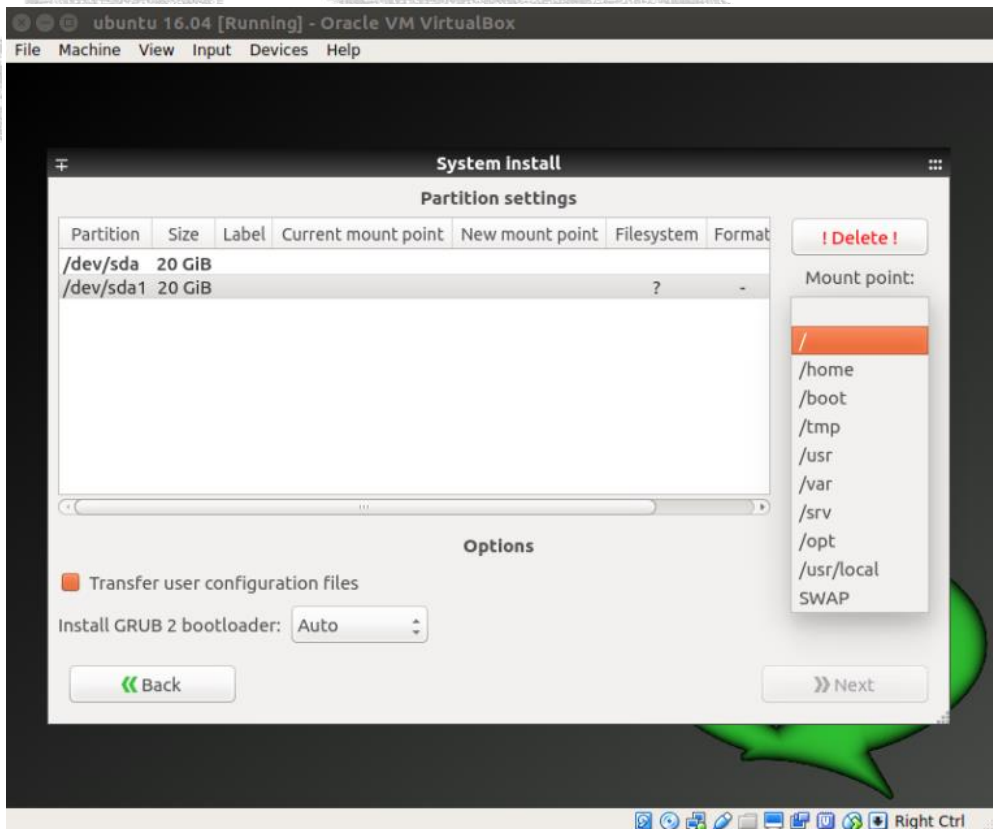




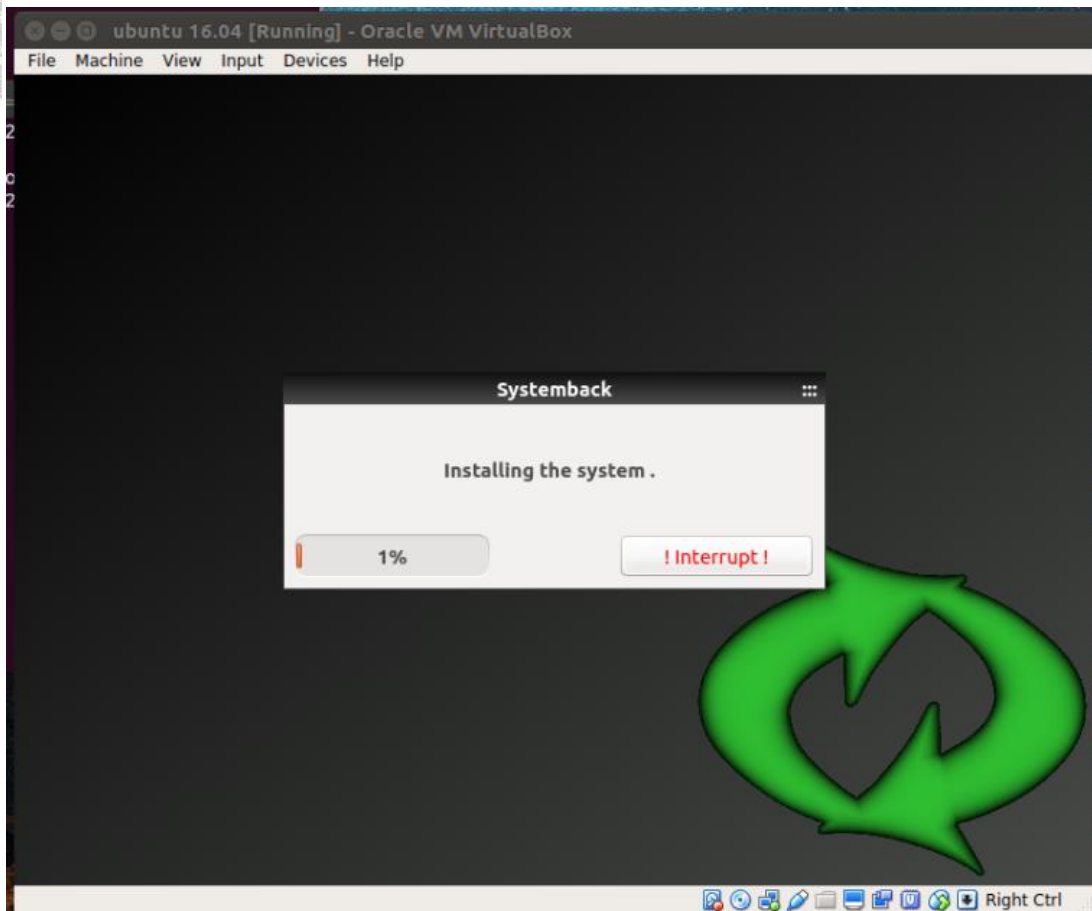
如上图类似设置，注意为了保证程序正常执行，请根据系统安装位置选择下述之一的设置，在小强主机上操作时：用户名一定要是 **xiaoqiang**，计算机名一定要是 **xiaoqiang-desktop** 在自己电脑上操作时：用户名设为 **robot**，计算机名设为 **bluewhale-desktop**

下一步进行分区，注意勾选上 **Transfer user configuration and data files**





等待安装完成就可以了

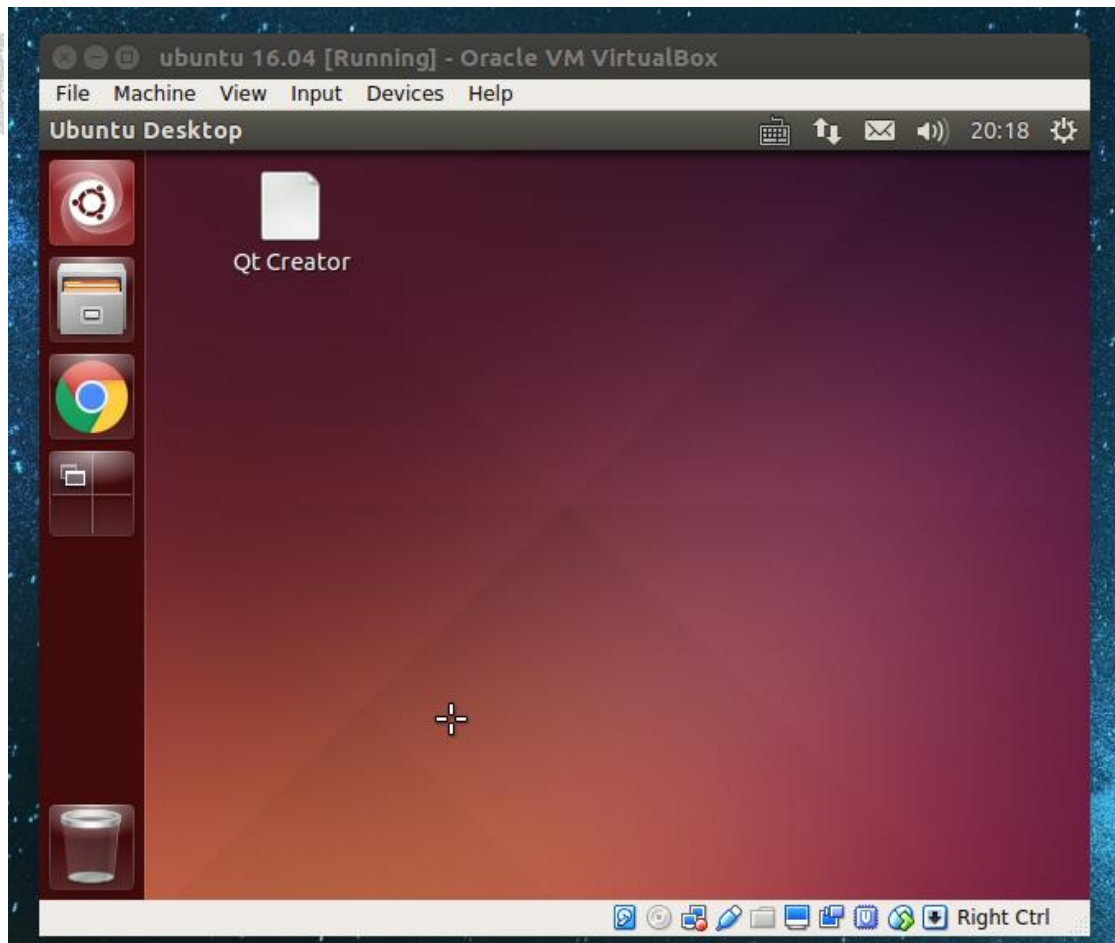


安装完成之后

安装后重启。安装完成之后可能会提示有一些错误，这是由于里面的一些残留文件导致的，可以把这些文件删除。在终端输入

```
sudo rm -rf /var/crash/*
```

然后再次重启，就可以了



在虚拟机中安装小强系统镜像的用户，请关闭开机启动项，避免与小强冲突

```
sudo service startup stop  
rosrun robot_upstart uninstall startup
```

Enjoy it

四、Ubuntu 设置静态 IP

在网上搜索设置静态 IP 出现的都是通过修改文件进行设置的。这样的方式很容易让电脑的网络不能使用。推荐使用下面的方法进行修改

IP 分配的基本方式

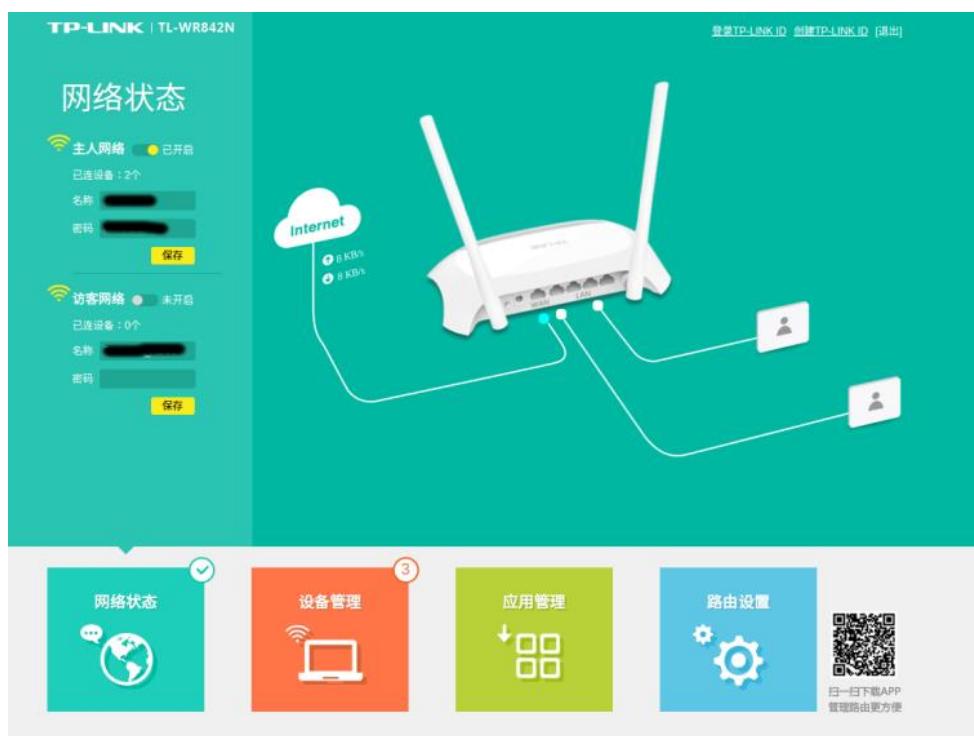
IP 分配的方式一般是由路由器决定的。路由器有 DHCP 方式和静态 IP 两种方式。DHCP 就是动态的分配 IP 方式。一般路由器默认的就是这种模式。在这种模式下电脑也可以设置自己的静态 IP。当然并不能保证一定会成功，比如自己设定的 IP 可能被别人占用。也有可能路由器由于迷之原因不给你分配自己设定的 IP。下面就具体说一下各种设置静态 IP 的方法。

设置静态 IP 的几种方式

1. 路由器设置法

通过路由器设置静态 IP 是最简单的方法。不过前提是你要有路由器的管理权限，路由器也要支持这种设置功能。

首先在浏览器中输入路由的地址，进入路由器的管理界面。



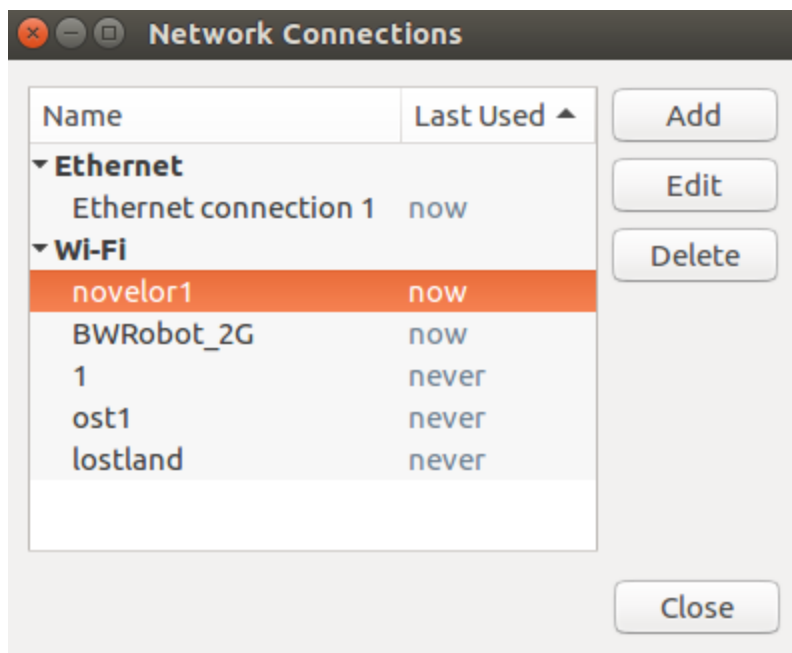
在应用管理中有 IP 和 MAC 绑定。不同的路由器设置界面不一样，你可以找一找这个功能的具体位置。



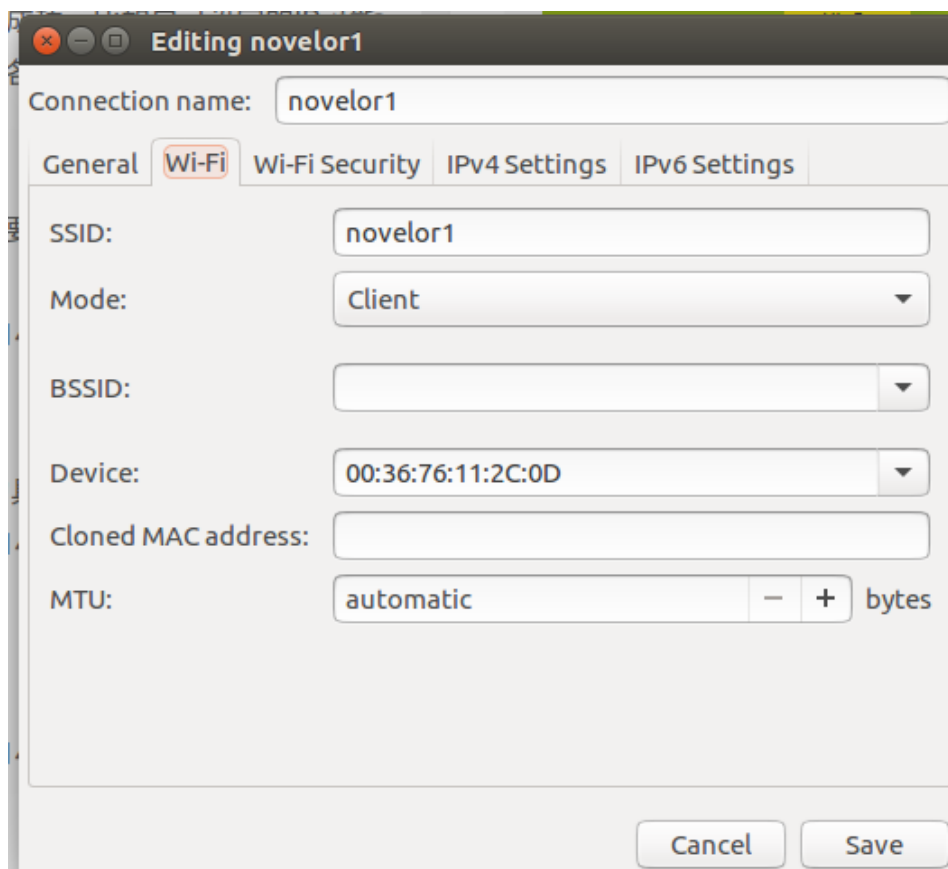
在这里就可以设置静态 IP 了



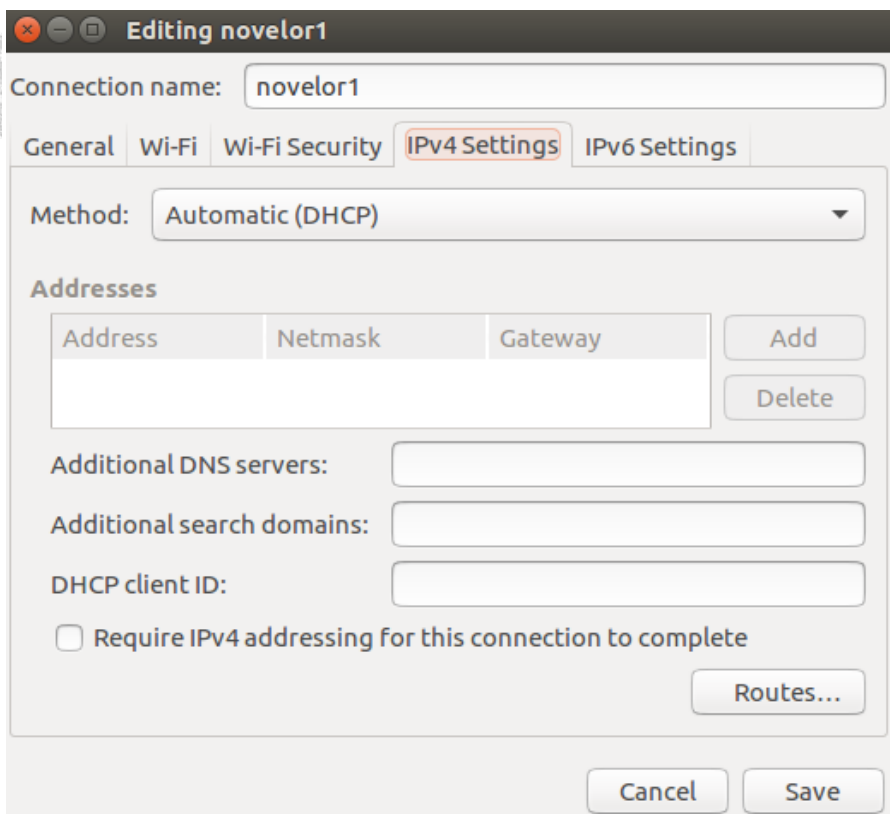
2. 在电脑上进行设置
电脑连接上网络后，打开电脑的网络管理器



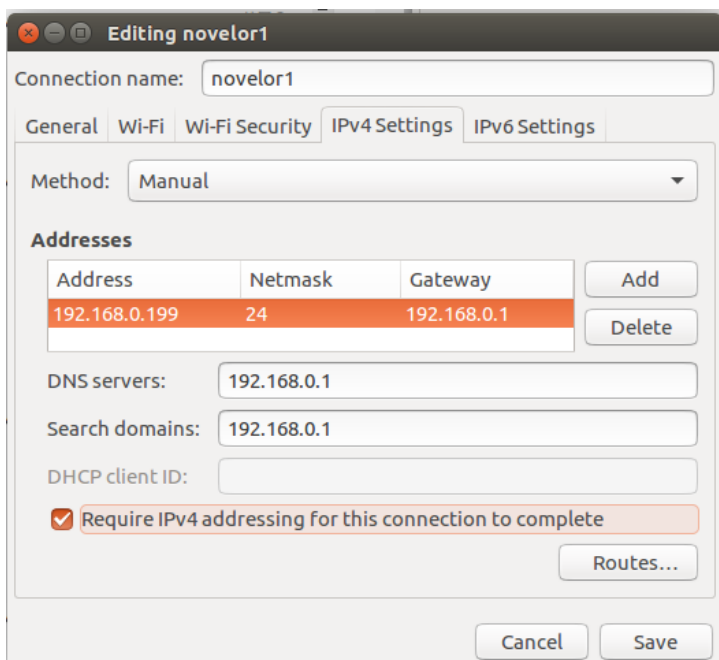
选中自己当前的网络链接，选择编辑



点击 IPv4 设置



按照下图这样进行设置



其中 192.168.0.199 是你想要设置的本机 IP, 要保证不能和局域网中的其他 IP 重复。192.168.0.1 是路由器的地址, 这个要根据自己的网络情况进行调整。设置完成后点击保存就可以了

然后断开网络重新连接 **一定要重新连接后才能生效**

```
randoms@nowhere:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr fc:aa:14:a1:af:f0
          inet addr:192.168.0.104  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::feaa:14ff:fea1:aff0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:109449 errors:0 dropped:0 overruns:0 frame:0
          TX packets:111052 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:53269931 (53.2 MB)  TX bytes:30415250 (30.4 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:27928 errors:0 dropped:0 overruns:0 frame:0
          TX packets:27928 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:8926946 (8.9 MB)  TX bytes:8926946 (8.9 MB)

wlan1     Link encap:Ethernet  HWaddr 00:36:76:11:2c:0d
          inet addr:192.168.0.199  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::b4e4:1ad2:575d:d856/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:468 errors:0 dropped:0 overruns:0 frame:0
          TX packets:293 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:85815 (85.8 KB)  TX bytes:49309 (49.3 KB)

randoms@nowhere:~$ █
```

输入 ifconfig 查看当前的网络信息，可以发现我的无线网络已经设置成刚才的那个 IP 了。

五、视觉导航路径编辑器使用教程

利用小强可以建立出周围环境的三维地图,但是如何利用这个地图实现视觉循迹呢?视觉导航路径编辑器就是为了实现这个功能而编写的。通过这个软件你可以在三维空间中标记你想要小强运动的轨迹。然后将生成的轨迹文件导出给小强,小强就能按照你标定的轨迹进行运动了。下面就详细介绍一下软件的使用方法。

1.安装

软件提供了 Ubuntu 的 deb 安装包可以在[这里](#)下载
下载完成后执行一下指令安装

```
sudo dpkg -i path-drawer_1.0.0_amd64.deb
```

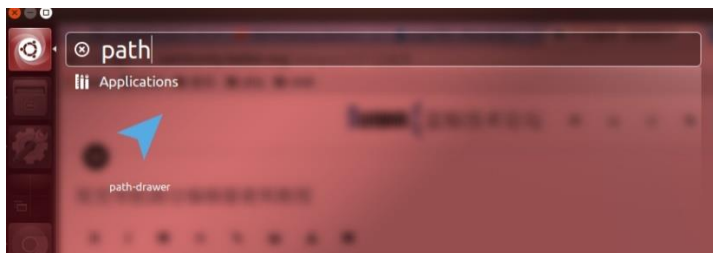
等待安装完成即可

2.建立视觉地图

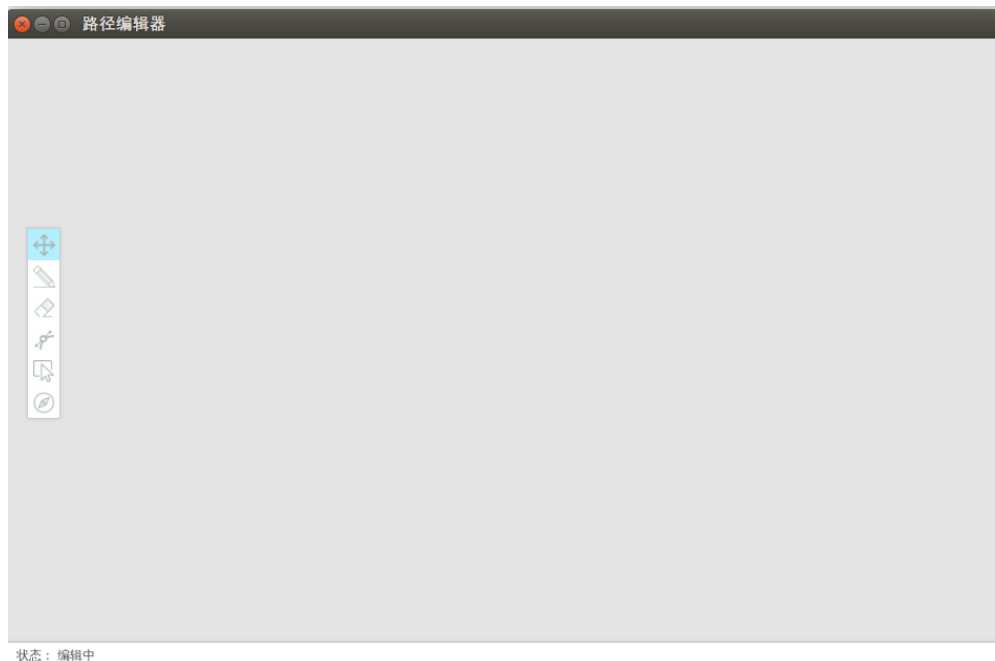
路径编辑器需要载入小强采集的空间数据才能够进行操作,详细的操作可以参考[这一篇](#)。地图信息默认存储在 /home/xiaoqiang/slamdb 文件夹内。

3.启动软件

安装完成后可以在 Ubuntu 的 Dash 菜单中找到名为 Path Drawer 的程序,点击启动即可。

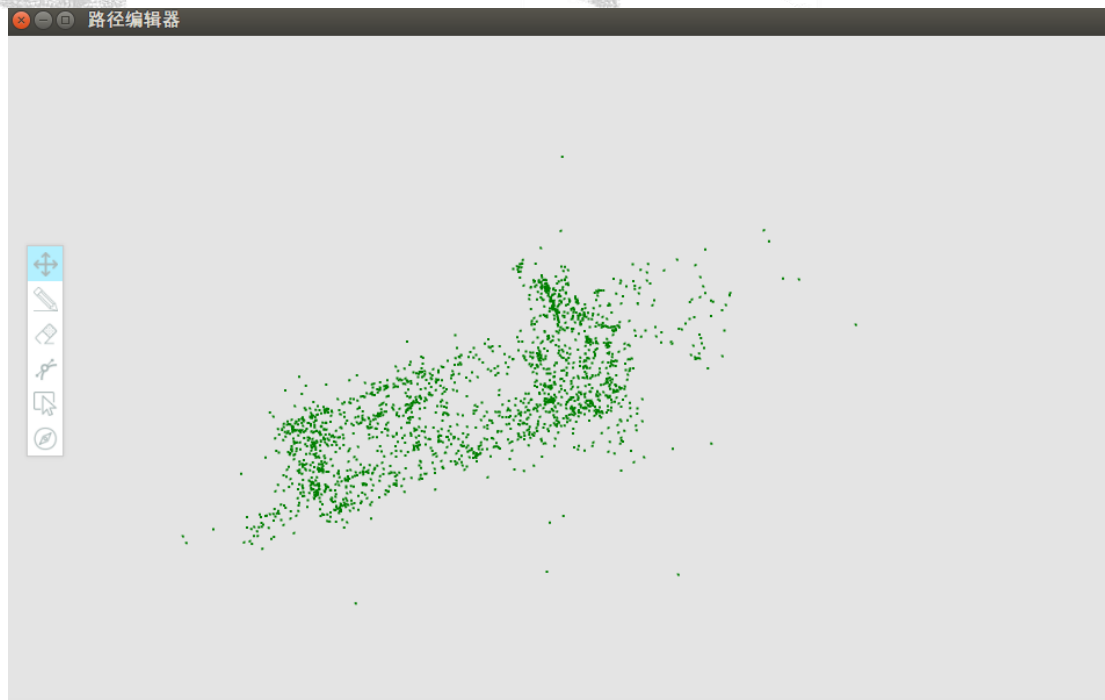


启动后的软件界面如图所示



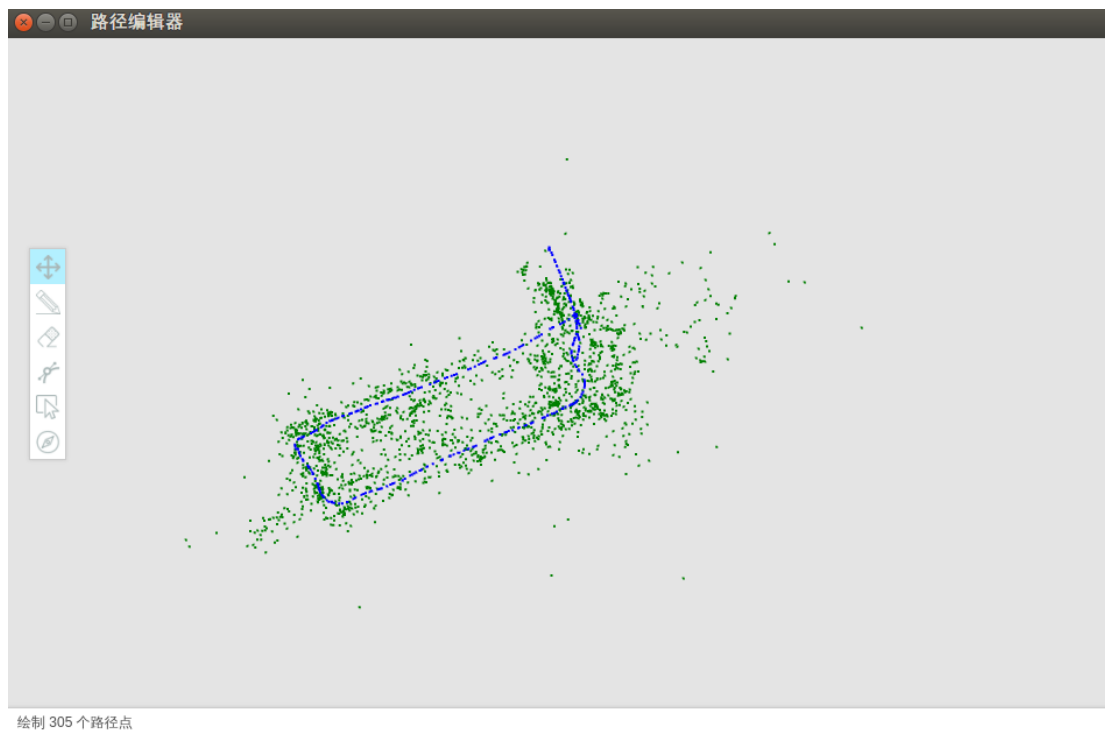
在左上角的菜单中选择文件->导入地图数据。在弹出的文件选择对话框中选择 `/home/xiaoqiang/slamdb/mappoints.bson` 文件。

成功导入后就能在软件中看到地图点的数据。这是从上向下的俯视图。



然后继续在左上角的菜单中选择文件->导入路径文件。在弹出的文件选择对话框中选择 `/home/xiaoqiang/slamdb/keyframes.bson` 文件。

成功导入后能在软件中看到之前小车行走的路径。



4. 绘制导航路线

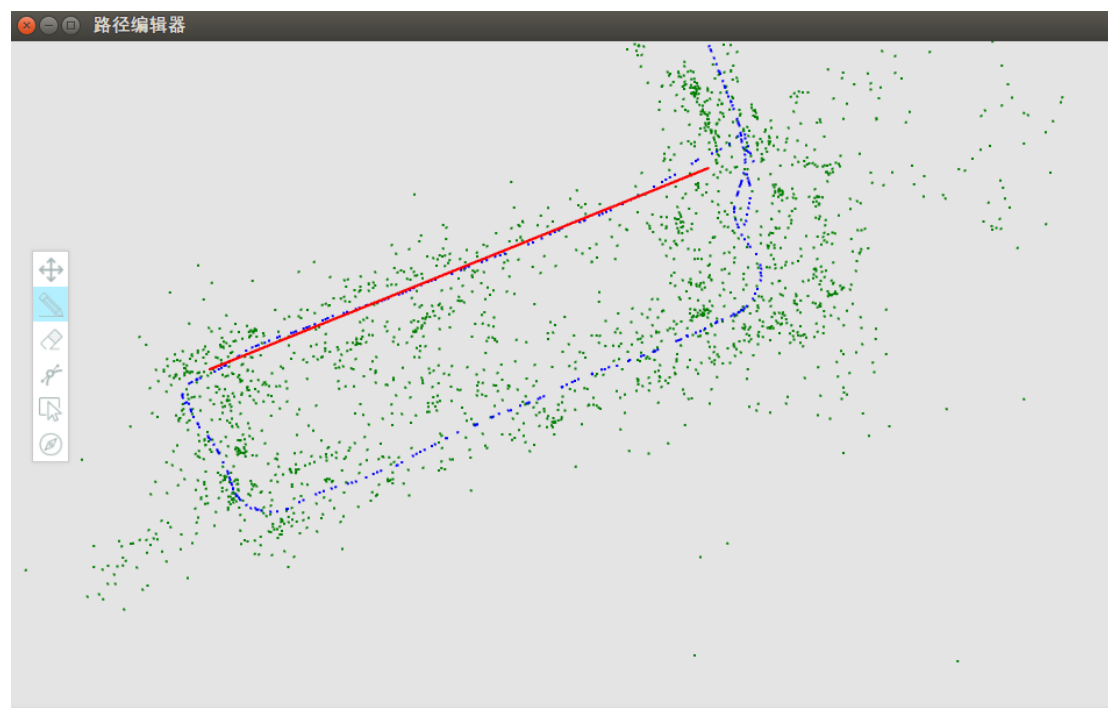
导航路线就是你想要小强行走的路径。当数据导出到小强后，小强就会按照你画的路径进行移动。下面介绍一下路径绘图工具的使用方法。

a. 基本操作

基本操作包括平移和缩放。如果鼠标左键拖动地图可以实现地图的平移。鼠标滚轴前后滚动可以实现地图的缩放，这在绘制路径的过程中非常的有用。对于对运动要求比较细致的地方可以放大后进行绘图。

b. 直线工具

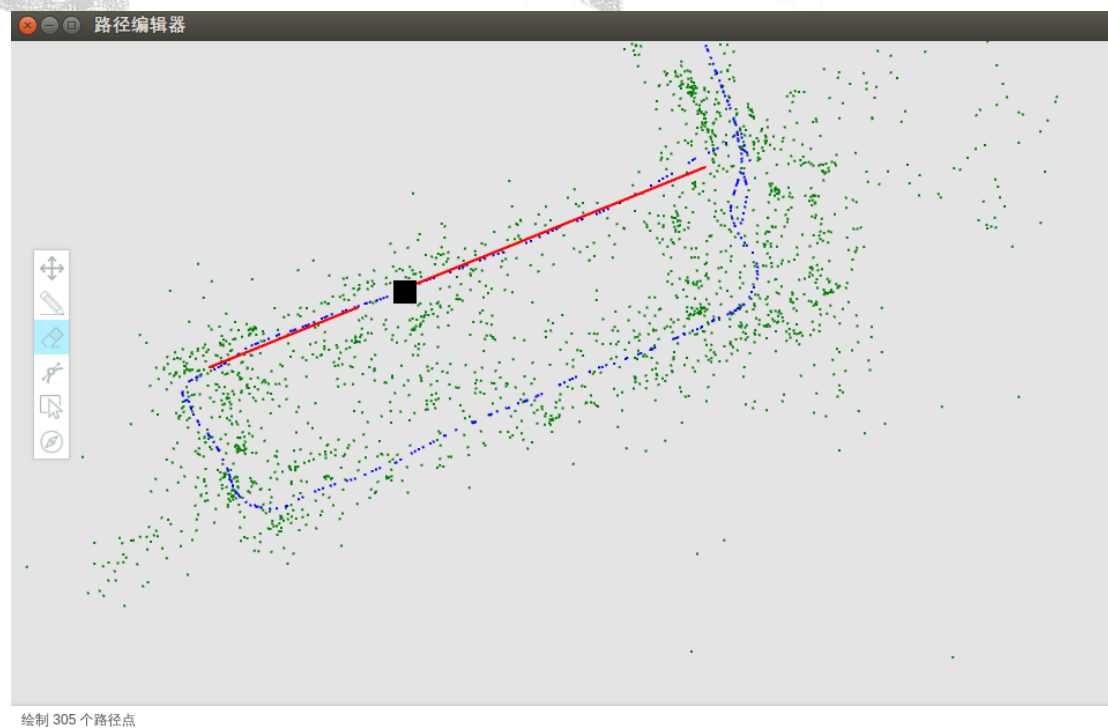
点击左侧工具栏里的铅笔一样的图标。这就是直线工具。鼠标左键点击图上任意一点，然后移动鼠标就会出现一条红色直线。移动鼠标到想要的终止位置，再次点击鼠标左键，一条直线就绘制完成了。在点击一次左键之后，点击右键就可以取消此次绘图。



绘制 305 个路径点

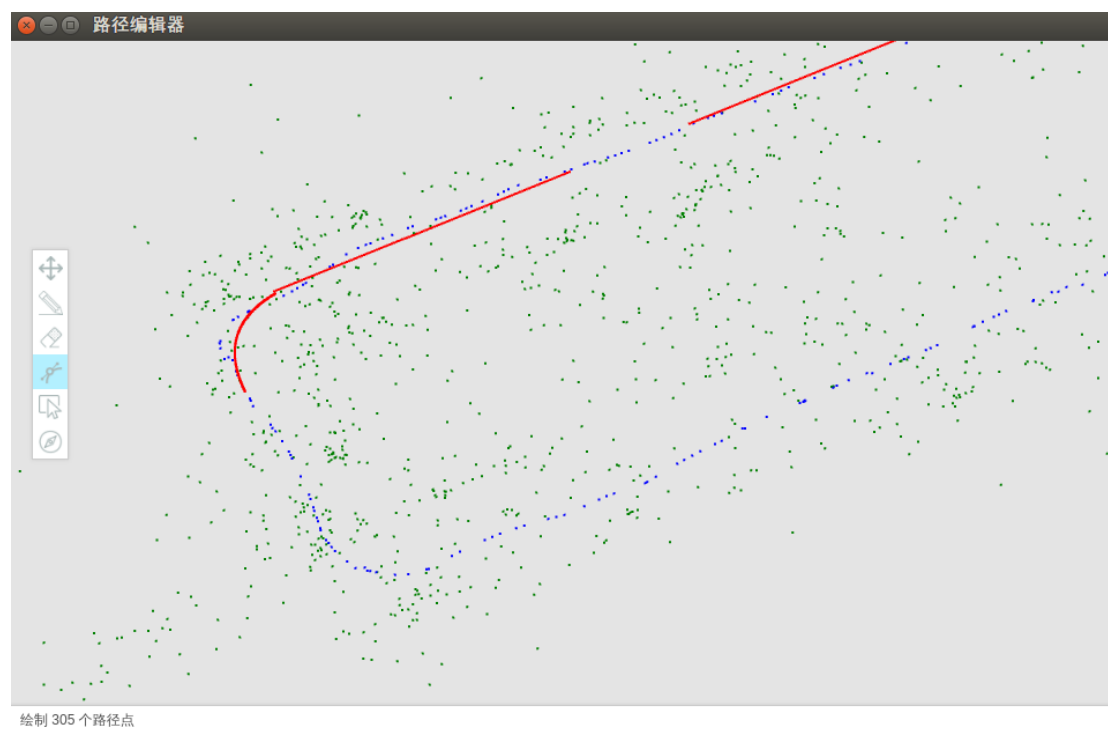
c. 橡皮擦工具

点击左侧工具栏中的橡皮擦工具，然后按下鼠标左键进行拖动就可以擦除之前绘制的点。



d. 曲线工具

点击左侧曲线工具，在曲线的起始点点击鼠标左键，然后在曲线的中间的再次点击一次鼠标左键，最后在曲线的结束点点击鼠标左键。这样一条曲线就绘制完成了。



e. 删除工具

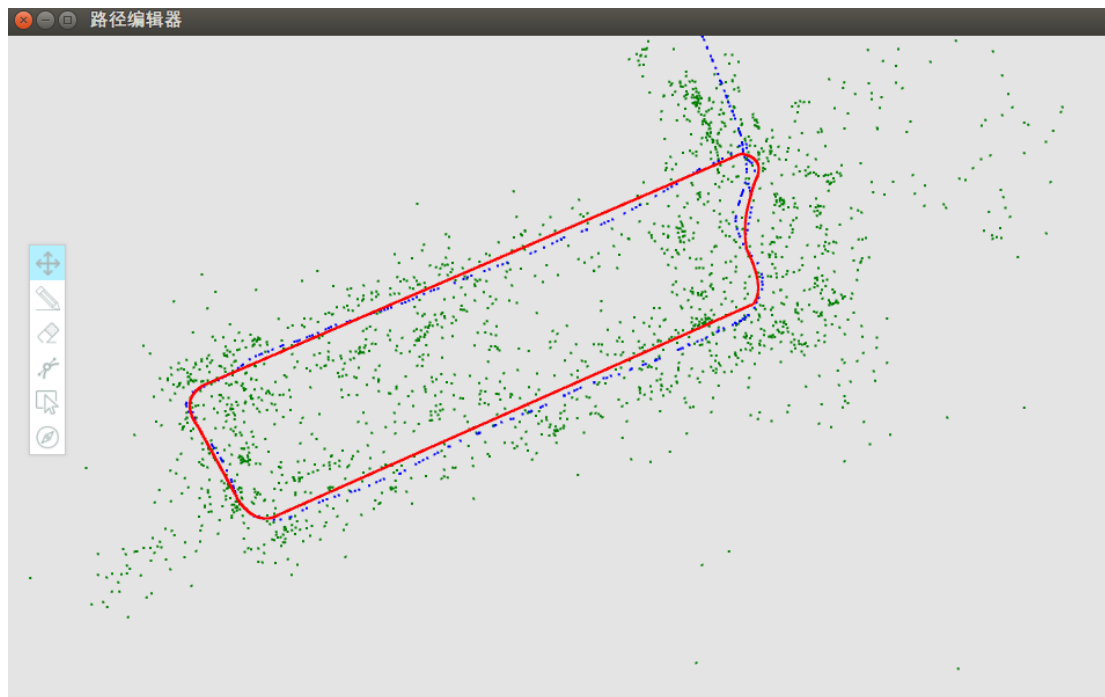
如果想要大范围的删除之前绘制的点，那么就可以利用这个删除工具。点击左侧的删除工具然后鼠标左键点击删除的起始点，可以看到在鼠标的移动过程中有一个矩形一直在跟随。再次点击鼠标左键就可以删除矩形选中的范围。右键可以取消选择。

利用这几个工具就可以绘制出小强的导航路径了。注意要尽量沿着原有的轨迹进行来画线，这样可以保证在运动过程中路线是畅通的。从绿色的地图点可以大致看出地形，根据这些信息画出运动所范围允许的点。

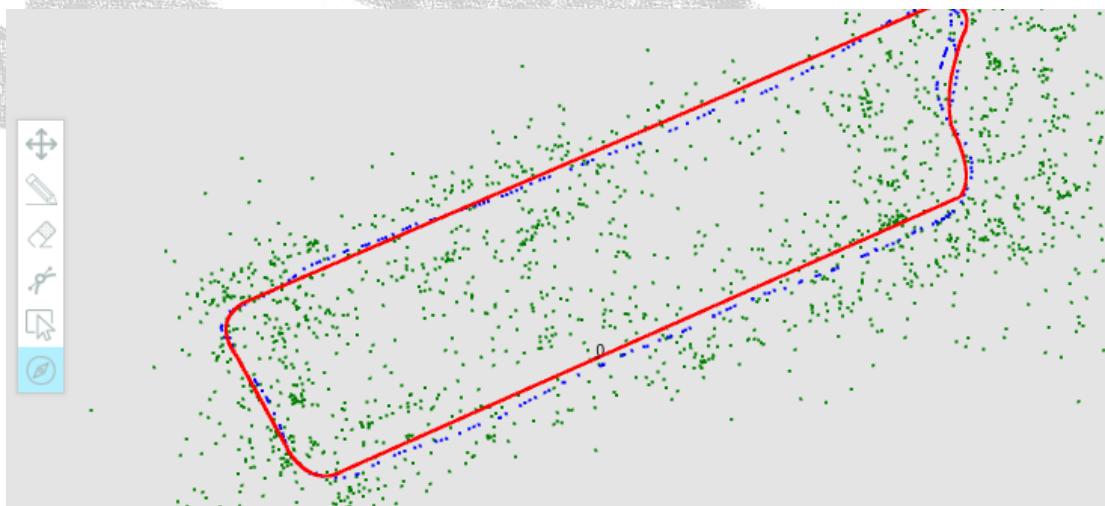
f. 设置导航关键点

对于比较复杂的图形可能运动的方式有很多种。比如一个 8 字形路径，小强可能先绕其中的一个圆运动，然后再绕另一个圆运动，也可以两个圆交叉的运动。所以很有必要指明小强运动的具体方式。

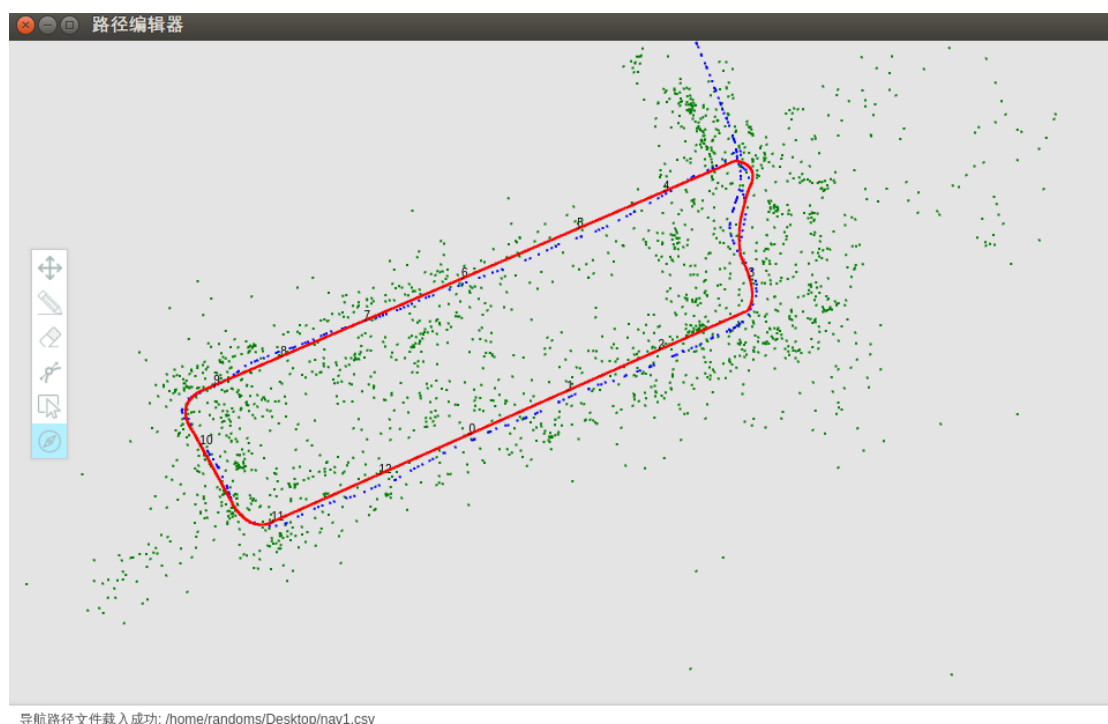
下面以一个圆形轨迹为例。在圆形轨迹中，小强可以顺时针运动，同时也可能是逆时针运动。



点击左侧工具栏最下面的导航点设置按钮。然后开始标记关键点。随意点击导航路线上的一个点，可以看到，在这个点上出现了一个 0。这就表明 0 号点已经被添加到处。



如果想要小强逆时针运动，就可以在右侧标记一个点。就这样依次把关键点加上



点击鼠标右键可以删除最近添加的一个导航点。同样也可以利用橡皮擦和删除工具来删除导航点。小强会按照关键点标记的顺序进行运动。

5. 导出数据

a. 导出导航路径文件

当导航路径绘制完成之后，在左上角的菜单中点击文件->导出导航路径文件，选择文件保存的位置即可。在导出文件后还可以从菜单中导入，进行二次编辑。

b. 导出导航关键点文件

当导航关键点绘制完成之后，在左上角的菜单中点击文件->保存导航点，选择文件的保存位置即可。在导出文件后同样也可以再次从菜单中导入，进行二次编辑。注意：只能在导航路径文件导入成功之后才能导入导航关键点。

导出的数据放入小强的对应文件夹内就可以开始视觉导航了。

六、小强的远程协助功能

为了为您提供更好的服务,在 2016 年 6 月之后发售的小强都默认安装了远程协助软件。通过这个软件我们的技术人员能够直接连接到您的小强,为您解决技术问题。当然您也可以通过程序协助去更加方便的远程遥控自己的小强。下面介绍一下小强的远程协助功能的具体使用方法:

1. 确认远程协助是否已经启动

a. 在终端执行 `htop`

```

randoms@nowhere: ~/Downloads
  1  [|||||] 15.4%  5  [|||||||||||||] 53.2%
  2  [|||||] 14.1%  6  [||] 1.3%
  3  [||||] 8.0%    7  [||] 0.0%
  4  [||||] 5.2%    8  [||] 1.3%
Mem[|||||||||||||] 4828/15947MB  Tasks: 228, 610 thr; 3 running
Swp[|] 0/0MB  Load average: 1.93 1.87 1.85
Uptime: 05:15:53

  NI  VIRT  RES  SHR  S  CPU%  MEM%  TIME+  Command
  0  761M  39352  2644  S  3.9  0.2  14:11.34  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:03.57  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:03.50  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:03.64  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:00.00  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:24.37  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:24.17  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:01.00  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  1.3  0.2  6:38.28  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  1.3  0.2  5:36.67  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:00.00  /usr/bin/cli ./SharpLink.exe
  0  761M  39352  2644  S  0.0  0.2  0:00.00  /usr/bin/cli ./SharpLink.exe

F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 SortBy F7 Nice -F8 Nice +F9 Kill F10 Quit

```

如果能够看到叫做 SharpLink 的线程正在执行说明远程协助已经开始执行了。

还可以通过更简单的方法确认程序的状态,执行下一指令

```
sudo service toxserver status
```

如果终端显示

```
toxserver start/running, process 1069
```

则说明程序已经正常执行了。

2. 远程连接小强

a. 查看自己的 ID

每台小强都有一个自己的 ID,请不要轻易的告诉别人。因为如果你没有更改默认密码,别人又知道你的 ID 的话,那么他就可以轻易的对你的小强进行操作。

查看 ID 的方法也非常简单,在终端执行

```
sudo service toxserver restart
```

```
tail -f ~/Documents/SharpLink/SharpLink/bin/Debug/server.log
```

此时终端的输出为

```
Time: 1467270721602,
```

```
ID:D70101972AB9B7E674290A25485B3752EDA236F65EF2C4AC7E738390DA61903565E68B8C43
```

```
1B
```

ID 后跟着的很长的字符串就是您的 ID。

如果您想要我们提供远程协助，只需要把这个 ID 告诉我们的工作人员就可以了。远程连接自己的小强。在记住自己的 ID 之后，只要您的小强连接上互联网，您就可以在任意地方随时的控制它，不会受到路由器，局域网的限制。

b. 首先在您的电脑上安装 SharpLink

这个软件是跨平台的，无论是 Linux 还是 Windows 都可以安装。安装的具体方法在项目的介绍里面。

c. 安装完成之后，在终端执行

```
sharplink 9999 你的 ID 127.0.0.1 22
```

这个指令会把本地 9999 端口和小强的 22 端口映射起来。只要连接本地的 9999 端口你就可以和小强的 22 端口相连了。当然你也可以把 9999 换成自己喜欢的端口。

此时在本地电脑上执行

```
ssh -p 9990 xiaoqiang@127.0.0.1
```

等待连接完成就可以控制小强了。

d. 打开和关闭远程协助

如果你想要关闭远程协助也是非常简单的。在终端删除对应的服务文件就可以了。

```
sudo service toxserver stop
```

```
sudo mv /etc/init/toxserver.conf /etc/init/toxserver.conf.back
```

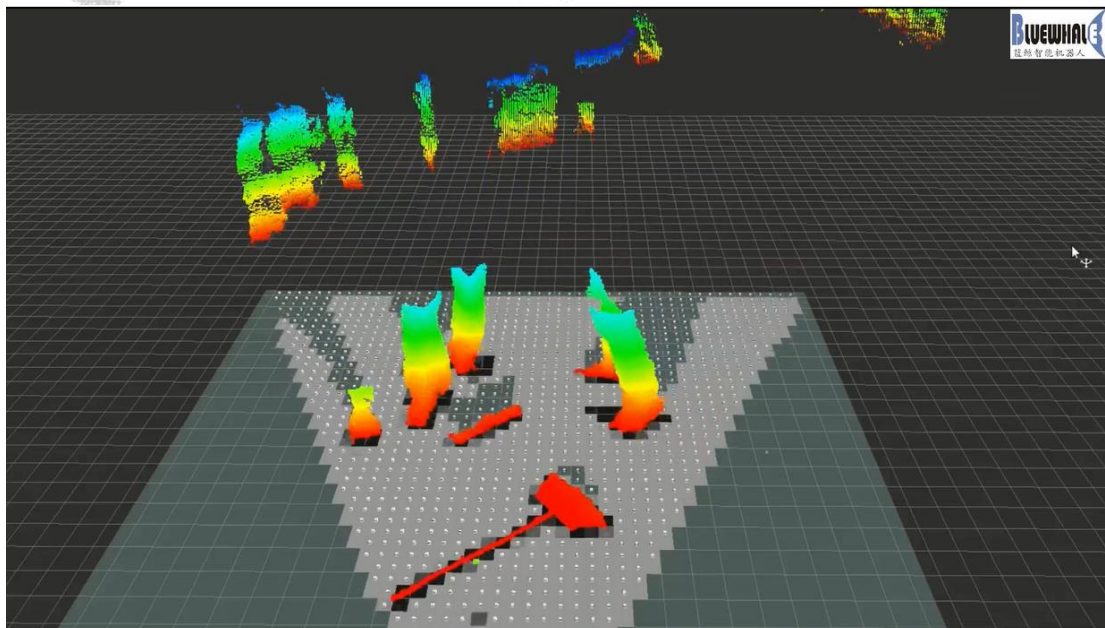
如果你想重新打开远程协助服务

```
sudo mv /etc/init/toxserver.conf.back /etc/init/toxserver.conf
```

```
sudo service toxserver start
```

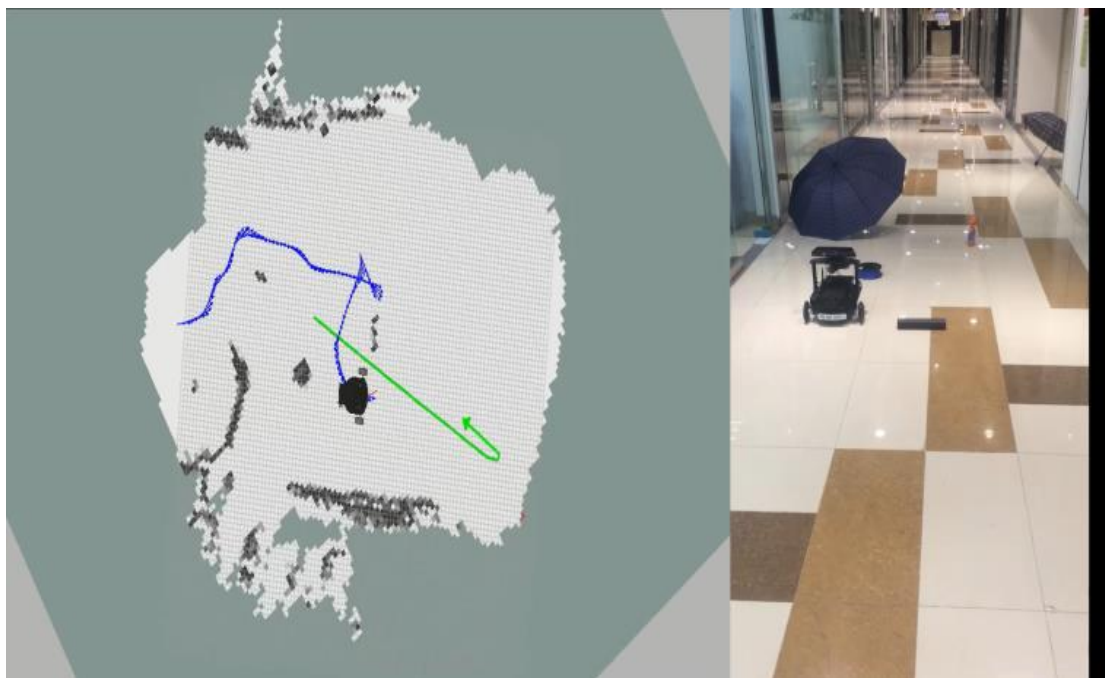
七、小强 ROS 机器人障碍物识别演示

下图是视频截图，完整演示视频请[点击这里](#)，平台测试代码请关注蓝鲸开源仓库



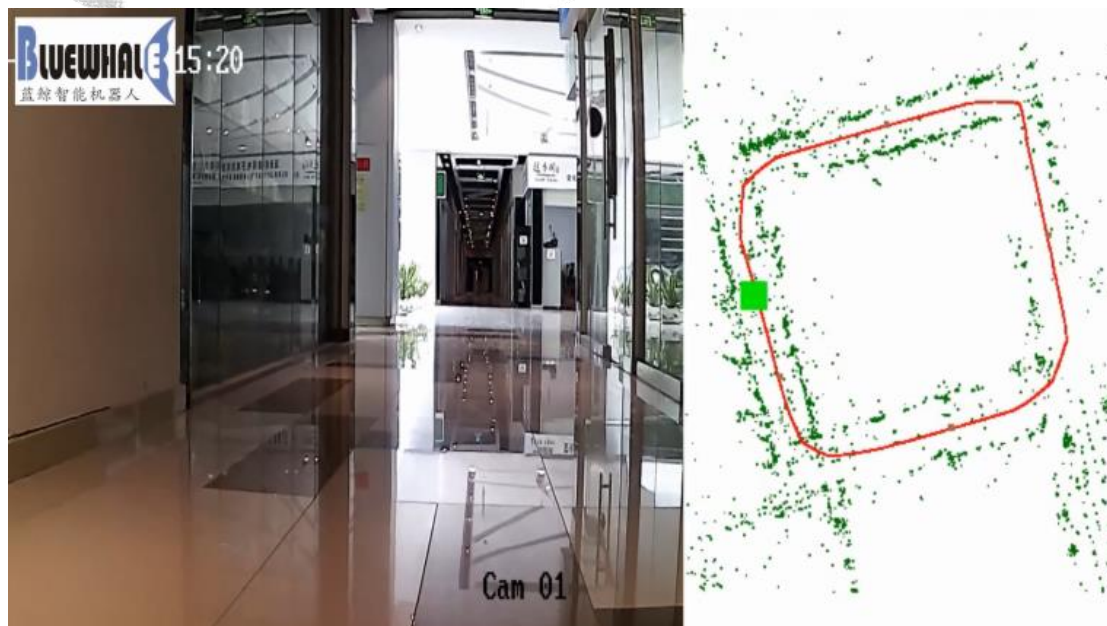
附上第二个小测试视频[点击观看](#)

附上自动驾驶测试视频[点击观看](#)



八、视觉导航在履带车中的运用

测试视频中的履带车采用了小强开发平台中的技术, 请[\[点击\]](#)观看完整视频

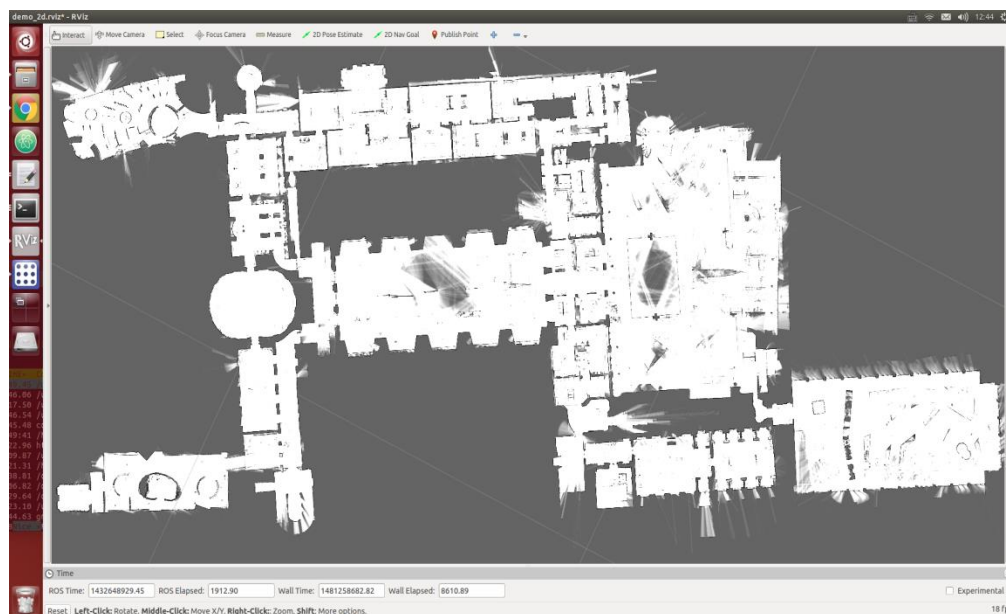


后台 app 界面截图:



九、Google 激光雷达 slam 算法 Cartographer 的安装及 bag 包 demo 测试

Cartographer 是 google 于 2016 年 9 月开源的一套激光雷达 slam 算法,精度和效果在业界处于领先水准。本文将演示在 ROS JADE 版中的安装使用方法。先上教程 demo 视频[点击查看](#)



操作步骤:

1. 安装依赖包

```
# Install the required libraries that are available as debs.
sudo apt-get update
sudo apt-get install -y \
  cmake \
  g++ \
  git \
  google-mock \
  libboost-all-dev \
  libcairo2-dev \
  libeigen3-dev \
  libgflags-dev \
  libgoogle-glog-dev \
```

```
liblua5.2-dev \  
libprotobuf-dev \  
libsuitesparse-dev \  
libwebp-dev \  
ninja-build \  
protobuf-compiler \  
python-sphinx
```

2. 安装 ceres solver

```
cd ~/Documents  
git clone https://github.com/BlueWhaleRobot/ceres-solver.git  
cd ceres-solver  
mkdir build  
cd build  
cmake ..  
make -j  
sudo make install
```

3. 安装 cartographer

```
cd ~/Documents  
git clone https://github.com/BlueWhaleRobot/cartographer.git  
cd cartographer  
mkdir build  
cd build  
cmake ..  
make -j  
sudo make install
```

4. 安装 cartographer_ros

```
cd ~/Documents/ros/src #请修改路径到自己的 ROS catkin 工作空间  
git clone https://github.com/BlueWhaleRobot/cartographer_ros.git  
cd ..  
catkin make
```

5. 安装已完成, 开始下载测试用的 bag 文件

点击下述链接下载文件, 保存到桌面

https://storage.googleapis.com/cartographer-public-data/bags/backpack_2d/cartographer_paper_deutsches_museum.bag

6. 启动 d e m o 演示, 正常可以看到 r v i z 启动并开始建图

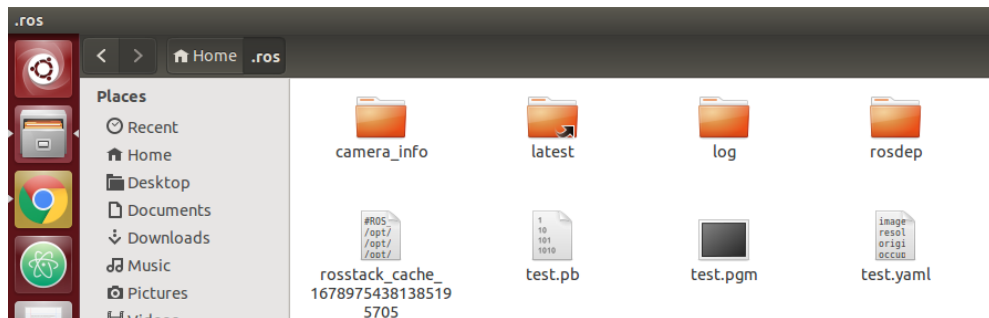
根据个人平台计算能力不同, 本 d e m o 完整运行时间一般为半个小时到 1 个小时之间

```
roslaunch cartographer_ros demo_backpack_2d.launch
```

```
bag_filename:=${HOME}/Desktop/cartographer_paper_deutsches_museum.bag
```

7. 保存地图, 结束测试

```
rosservice call /finish_trajectory "stem: 'test'"
```



现在在 home 目录下的 .ros 文件夹内会生成建立的地图文件, 这两个文件 (pgm 和 yaml) 在 ros 中的 map_server 中可以加载使用

十、原装和国产 ps3 手柄 ros 驱动程序

ps3 手柄的 ros 驱动程序为 joystick_drivers 包中的 ps3joy,这份驱动对索尼原装手柄支持较好,但是对国产 ps3 手柄的支持存在问题。我们在 ps3joy 的基础上进行了修改,增加了一个 ps3joyfake_node.py 脚本作为国产手柄的驱动,包源代码地址在这里:https://github.com/BlueWhaleRobot/joystick_drivers.git。下文以小强为例,演示这个包的安装步骤和简略的使用方法

安装步骤

ssh 进入小强 ros 工作空间,下载源码后编译,完成安装

```
ssh xiaoqiang@192.168.xxx.xxx
cd Documents/ros/src/
git clone https://github.com/BlueWhaleRobot/joystick_drivers.git
cd ..
catkin_make
如果提示下列错误
error spnav.h no such file
先安装下面这个包,然后重新执行 catkin_make
sudo apt-get install libspnav-dev
```

快速使用方法

ps3joyfake_node.py 启动后,它会将蓝牙接收的手柄按键数据转换成标准的 joy msg,同时以 /joy 为话题在 ros 中发布,即 ps3joyfake_node.py 一个文件相当于 ps3joy.py + joy_node 两个文件,在实际使用中不必再开启 joy_node 节点。

1. 将手柄与 usb 蓝牙适配器进行绑定,只需绑定一次,下次直接跳过这个步骤

将手柄通过 usb 数据线接入主机,usb 蓝牙适配器也插上主机

```
sudo bash
roslaunch ps3joy sixpair
此时会得到类似下图的输出,current 和 setting 的 mac 地址一样说明绑定成功
Current Bluetooth master: 00:22:b0:d0:5a:09
Setting master bd_addr to 00:22:b0:d0:5a:09
如果出现下述错误
Current Bluetooth master: 00:1b:dc:00:07:3c
Unable to retrieve local bd_addr from `hcitool dev`.
Please enable Bluetooth or specify an address manually.
先运行 hciconfig hci0 reset
如果运行 hciconfig hci0 reset 出现错误
Can't init device hci0: Operation not possible due to
运行 rfkill unblock all 然后运行 hciconfig hci0 reset
重新运行 roslaunch ps3joy sixpair
```

绑定设置完成,断开手柄与主机的 usb 连接

```
ctrl+D 退出 root 模式
```

2. 将手柄与 usb 蓝牙适配器配对

确保蓝牙接收器已经插入主机 usb 口

```
sudo bash
```

```
roslaunch ps3joy ps3joyfake_node.py
```

正常会出现下面的提示

```
root@xiaoqiang-desktop:~# roslaunch ps3joy ps3joyfake_node.py
```

```
No inactivity timeout was set. (Run with --help for details.)
```

```
Waiting for connection. Disconnect your PS3 joystick from USB and press the pairing button.
```

如果提示下列错误

```
ImportError: No module named bluetooth
```

请先安装下列包,然后重新运行

```
sudo apt-get install libbluetooth-dev
```

```
sudo pip install PyBluez
```

按一下下图中的手柄配对键



配对成功的话,手柄会震动一下,同时上面的窗口会输出类似下面的结果

```
root@xiaoqiang-desktop:~# roslaunch ps3joy ps3joyfake_node.py
```

```
No inactivity timeout was set. (Run with --help for details.)
```

```
Waiting for connection. Disconnect your PS3 joystick from USB and press the pairing button.
```

```
Connection activated
```

配对完成

3. 查看手柄输出

新开一个窗口，打印按键输出

```
rostopic echo /joy
```

正常会出现下面的类似结果

```
header:
```

```
seq: 297
```

```
stamp:
```

```
secs: 1488877867
```

```
nsecs: 535818099
```

```
frame_id: ''
```

```
axes: [0.0, 0.0, 0.0, 0.0, -1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.21316899359226227, 0.0]
```

```
buttons: [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

从上面可以看到第 4 个 button 被按下了

4. 启动相关的 joy msg 处理节点

注意不要再启动 joy_node

以小强为例,启动下列 launch 文件后就可以遥控小强移动了

```
roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
```

十一、升级软件包以支持小强图传遥控 app

注：本教程适用于 2017 年 3 月份之前收到小强的用户，2017 年 3 月份之后的用户请跳过本教程，直接参考这两篇帖子的使用方法——[小强手机遥控 APP 安卓版](#)、[小强图传遥控 WINDOWS 客户端](#)

小强手机遥控 app 需要小强开机自动启动一些配套服务后才能使用，下文将演示升级小强主机上的软件包地具体步骤，完成这些步骤后重启小强主机即可开始使用手机端 app。

1. SSH 登陆小强主机

```
ssh xiaoqiang@192.168.xxx.xxx -X
```

2. 进入小强主机 ROS 工作空间，升级替换两个软件包

```
cd Documents/ros/src/  
rm -rf system_monitor  
rm -rf startup  
git clone https://github.com/BlueWhaleRobot/system_monitor.git  
git clone https://github.com/BlueWhaleRobot/startup.git  
cd ..  
catkin_make
```

3. 更新小强开机启动项

```
roslaunch robot_upstart uninstall startup  
roslaunch robot_upstart install startup/launch/startup.launch
```

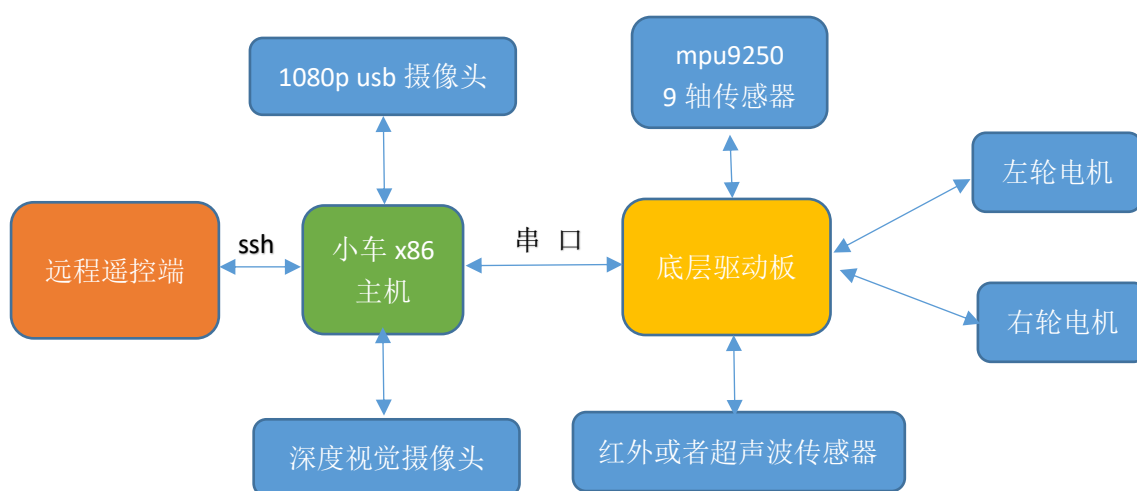
4. 重启小强主机，完成升级

```
sudo shutdown -r now
```

十二、附件

1. 小车系统框架图

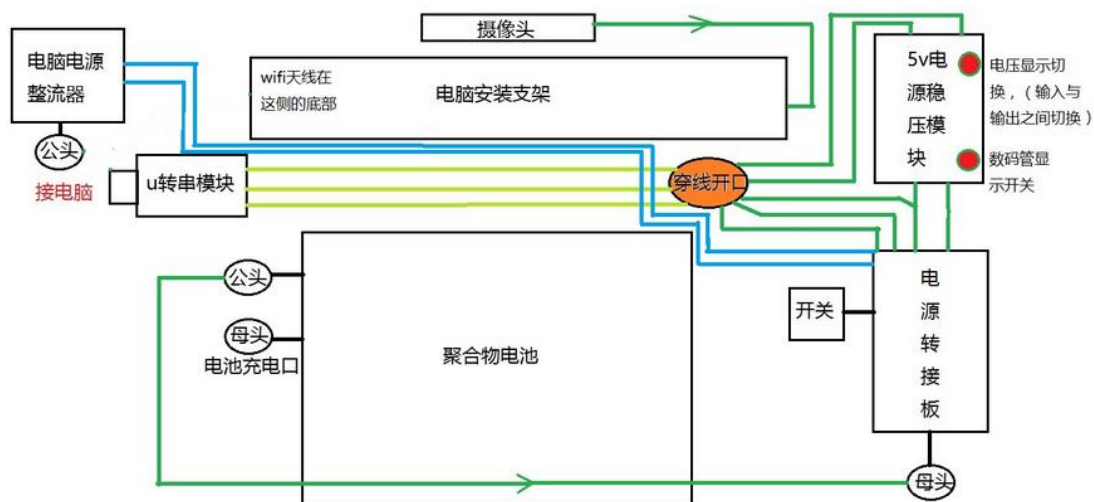
小车主机基本配置为 intel i7 4560U、64gb 固态硬盘、8gb 内存。20AH 聚合物电池，整机续航 7 小时以上。小车移动速度最大 0.8m/s，电机自带编码器配有速度控制，自适应多种承重负载和地面路况。



2. 电气布线图

小强底盘上的电线头部都带有线标，各种模块 pcb 上也有引脚标识，接线时只需要将线标与引脚标识对应起来就可以了，对应关系是名字一样。

这是小强底盘上端面俯视图：



3. 小强电脑与 stm32 底层通讯协议

串口波特率为 115200，8 个数据位，1 个停止位，无奇偶校验。

3.1 电脑下发指令

方向运动指令都是 6 个字节的无符号 byte 数组

a. 前进

0xcd	0xeb	0xd7	0x02	0x66	0xXX
包头	包头	包头	命令长度	前进指令	速度大小，数值范围为 0 到 100

b. 后退

0xcd	0xeb	0xd7	0x02	0x62	0xXX
包头	包头	包头	命令长度	后退指令	速度大小，数值范围为 0 到 100

c. 左转

0xcd	0xeb	0xd7	0x02	0x63	0xXX
包头	包头	包头	命令长度	左转指令	速度大小，数值范围为 0 到 100

d. 右转

0xcd	0xeb	0xd7	0x02	0x64	0xXX
包头	包头	包头	命令长度	右转指令	速度大小，数值范围为 0 到 100

e. 停止

0xcd	0xeb	0xd7	0x02	0x73	0xXX
包头	包头	包头	长度	前进指令	制动量大小，数值范围为 0 到 100

单个电机独立控制指令是 13 字节的

0xcd	0xeb	0xd7	0x09	0x74	0xXX	0xXX	0xXX	0xXX	0xXX	0xXX	0xXX	0xXX
包头	包头	包头	长度	类型	右轮电机指令	左轮电机指令	保留	保留	控制量大小	控制量大小	保留	保留

电机指令有三种状态，‘F’：向前，‘B’：向后，‘S’：刹车

控制量范围是 0 到 100

例如：

tSSSS0000 翻译成 hex 为 cd eb d7 09 74 53 53 53 53 00 00 00 00 表示让四个电机全部以 0% 的制动量刹车

对于小强硬件，只有前两个电机有效，第一个电机对应右轮，第二个电机对应左轮

3.2 小车上传指令

小车状态上传数据包格式：包头+长度+内容

包头：为 3 个 u8 字符：205 235 215

长度：1 个 u8 字符，长度不包括包头和长度本身字符，当前数据包长度为 90, 这个长度包括字符串结束符 0x00

内容：由 27 个 4 字节小端模式二进制表示的数字组成，数字之间用空格 0x20 分开。

小车坐标系为右手系，原点在两轮轴中心，小车正前方为 x 轴正方向，右轮指向左轮方向为 y 轴正方向方向。

完整数据包内容构成一个 c 语言结构体，结构体具体构成如下所示：

```
typedef struct {
    int status;//小车状态, 0 表示未初始化, 1 表示正常, -1 表示 error
    float power;//电源电压【9 13】v
    float theta;//方位角, 【0 360】°
    int encoder_ppr;//车轮 1 转对应的编码器个数
    int encoder_delta_r;//右轮编码器增量, 个为单位
    int encoder_delta_l;//左轮编码器增量, 个为单位
    int encoder_delta_car;//两车轮中心位移, 个为单位
    int omga_r;//右轮转速 个每秒
    int omga_l;//左轮转速 个每秒
    float distance1;//第一个超声模块距离值 单位 cm
    float distance2;//第二个超声模块距离值 单位 cm
    float distance3;//第三个超声模块距离值 单位 cm
    float distance4;//第四个超声模块距离值 单位 cm
    float IMU[9];//mpu9250 9 轴数据
    unsigned int time_stamp;//时间戳
}UPLOAD_STATUS;
```

