

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

Table of Contents

Introduction	1.1
Quick Start	1.2

Xiaoqiang ROS tutorial

1. Basic operation introduction and remote control in LAN	2.1
2. Bluewhale Robot Open source code repository usage and ROS startup task configuration	2.2
3. Displaying Xiaoqiang Robot Model in rviz	2.3
4. Inertial navigation test	2.4
5. Xiaoqiang remote control app for Android	2.5
6. Xiaoqiang remote control Windows client	2.6
7. Use PS3 joystick to control xiaoqiang	2.7
8. Kinect v1 ROS driver test and installation	2.8
9. Use rostopic to control kinect tilt angle	2.9
10. Use kinect for autonomous mobile and obstacle avoidance	2.10
11. Kinect follow package turtlebot_follower	2.11
12. Display point cloud for kinect2	2.12
13. Rplidar A2 LiDAR useage and set udev rules for serial devices for xiaoqiang	2.13
14. Using rplidar A2 with gmapping	2.14
15. AMCL navigation test	2.15
16. Large-scale lidar slam and real-time loop closure test	2.16
17. Using ORB_SLAM2 to create a three-dimensional model of the environment	2.17
18. 3D modeling using DSO_SLAM	2.18
19. Usage of NLlinepatrol_planner	2.19
20. Get vision odometer and display the xiaoqiang track in the rviz	2.20
21. Get USB camera 30fps 1080p image stream and 120fps VGA resolution image stream	2.21
22. Operating 6 DOF robotic arm	2.22
23. ROS introductions	2.23

Maintenance

Maintenance	3.1
Xiaoqiang Bottom Driver Firmware Download and Upgrade Methods	3.2

Upgrade chassis ros driver package xqserial_server	3.3
Recalibrate the chassis IMU	3.4
Bluewhale ROS system image released	3.5

Other Tutorials

How Ubuntu sets static IP	4.1
Xiaoqiang's remote assistance function	4.2
Google lidar slam algorithm Cartographer installation and bag package demo test	4.3
Install VNC on Ubuntu	4.4
PS3 joystick ROS driver	4.5
Cartographer install and demo	4.6
Visual Navigation Path Editor Tutorial	4.7

- [Introduction](#)

Introduction

This is the GitHub version of the Xiaoqiang ROS(Robot Operating System) robot user manual. This file is automatically generated by the tutorial from community. This manual contains all the necessary information for the installation and use of Xiaoqiang, and you can learn how to use it by reading this manual.

The following tutorials focus on the various functions of using Xiaoqiang and do not include the tutorial of ROS. If you are a beginner user of ROS, follow the tutorials below while learning the basics of ROS. Instead of simply following the tutorial to enter the instructions, it is recommended to try to understand the specific meaning of each command. In case you encounter problems, please try to find out the cause of your problems on your own. This can be done through searching the problem on [Google](#) or discussing on forums such as [StackOverflow](#).

If you encounter a tutorial or a place where ROS does not understand, you are welcome to ask questions in our [ROS Slack Group](#).

- [xiaoqiang quick start guide](#)
 - [Quick start](#)
 - [Start Using](#)
 - [Set up the network](#)
 - [Product assembly](#)
 - [State check](#)
 - [remote control](#)
 - [Software overall structure and description](#)
 - [ROS Primer](#)

xiaoqiang quick start guide

Quick start

Start Using

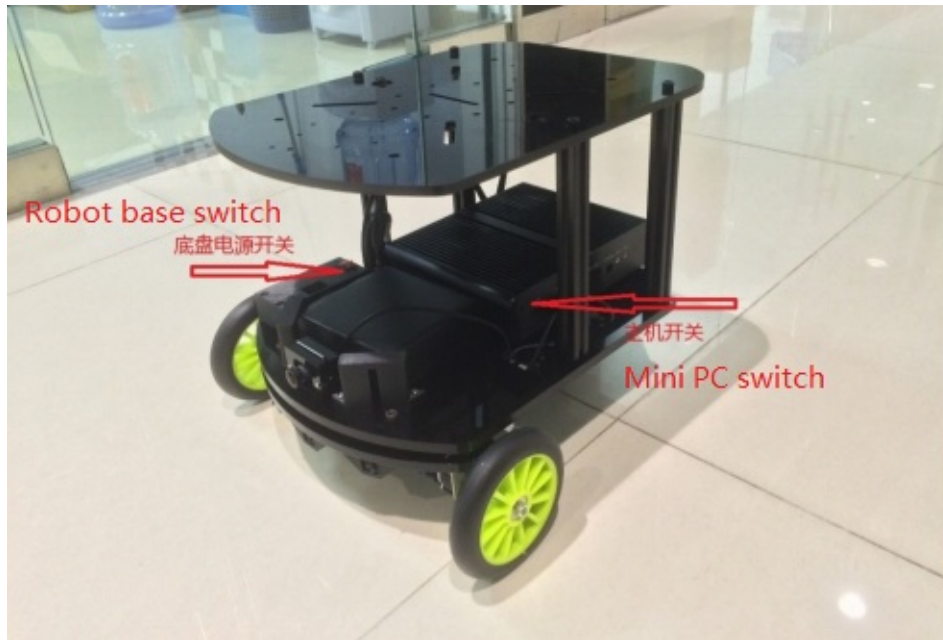
This chapter describes how to quickly use the platform. If you are not familiar with the ROS system and Ubuntu system, please read the [first tutorial of the second chapter](#).

Set up the network

First, connect monitor to Xiaoqiang's host pc (Xiaoqiang Pro users please use the included HDMI to VGA adapter to connect, Xiaoqiang Mini users please use vga directly) 。 Xiaoqiang's default password is `xiaoqiang` , Please change the default password。 Enter the OS and set up Xiaoqiang's wifi network connection。 It is recommended to set up the router so that Xiaoqiang can use static ip, which is convenient for future connections. [Setting up static IP tourial](#).

Product assembly

The main part of Xiaoqiang is shown in the figure below.



Place the battery flat on the free area in front of the host (the battery is blocked by two small black blocks). Connect the platform power cord according to the wire label. Installing computer host wifi antenna. USB camera and platform USB communication cable can be connected to the host's USB interface

State check

After the assembly is complete, you can start using Xiaoqiang. Turn on the Xiaoqiang host computer switch and wait for the host blue light to turn on. Xiaoqiang left power data display is normal (The normal range of battery voltage cannot be lower than 10V, The normal range of battery voltage can not be less than 10V or it will automatically shut down)

Remote connection through ssh, where xxx.xxx.xxx.xxx is Xiaoqiang's IP

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

Check that the program is working properly and execute the following commands

```
rostopic list
```

Normally, all current ROS topics will be displayed.

```
/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
```

If the topic is not displayed properly, you can try to restart the service.

```
sudo service startup restart
```

Check system status

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

If it is normal, the display is as follows

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

Where imageStatus indicates whether the camera is working properly. odomStatus indicates whether the underlying driver is working properly. The three Orb-related variables are visual navigation-related states that can be temporarily ignored (If you are interested in this area, you can communicate in the forum) . brightness is the brightness of the camera. Power is the voltage value of the current battery, If the voltage cannot be read, 0 is displayed.

remote control

Connect through ssh

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

Start remote control program

```
roslaunch nav_test control.py
```

Now you can use the arrow keys to control the movement of Xiaoqiang. Spacebar is stopped. Ctrl + C exits the program.

Software overall structure and description

Xiaoqiang's software is built on ROS. The program mainly contains the driver, navigation algorithm and slam algorithm. For robots, the software is a whole, and the dependence of each part is relatively high. For ease of management, The basic functions in the system are operated by one service. That is, the startup service mentioned earlier. This service starts the driver package and camera. Startup package is located

`/home/xiaoqiang/Documents/ros/src/startup` . This service is automatically started when the system starts. If you want to change the content of this service, you can modify the launch file in this package. For detailed operations, please refer to [here](#). The system's navigation program uses the navigation package that comes with ROS. However, the navigation parameters are modified according to Xiaoqiang's own parameters. Navigation parameters have a great influence on the performance of navigation. You can also try to modify the parameters yourself to improve performance. The slam algorithm uses ORB_SLAM. This algorithm is currently not available in the actual production environment, but its use performance is also very good. The last is some tool software. For example, `nav_test control.py` related to control platform movement. , System monitor that shows the system status.

ROS Primer

[Learning ROS for Robotics Programming - Second Edition.pdf](#). This tutorial is very basic and comprehensive, Although taking the Hydro version as an example, But it is also fully compatible with the Kinetic version, In the code example in the book, just replace the string `hydro` with `kinetic` . Please read the second and third chapters of this book in detail.

- [xiaoqiang tutorial \(1\) Basic operation introduction and remote control in LAN](#)
 - [Basic operation introduction and remote control in LAN](#)
 - [before the start](#)
 - [1.Configure the Xiaoqiang network](#)
 - [2. Local configuration](#)
 - [FAQ](#)

xiaoqiang tutorial (1) Basic operation introduction and remote control in LAN

[Xiaoqiang Homepage](#)

Basic operation introduction and remote control in LAN

before the start

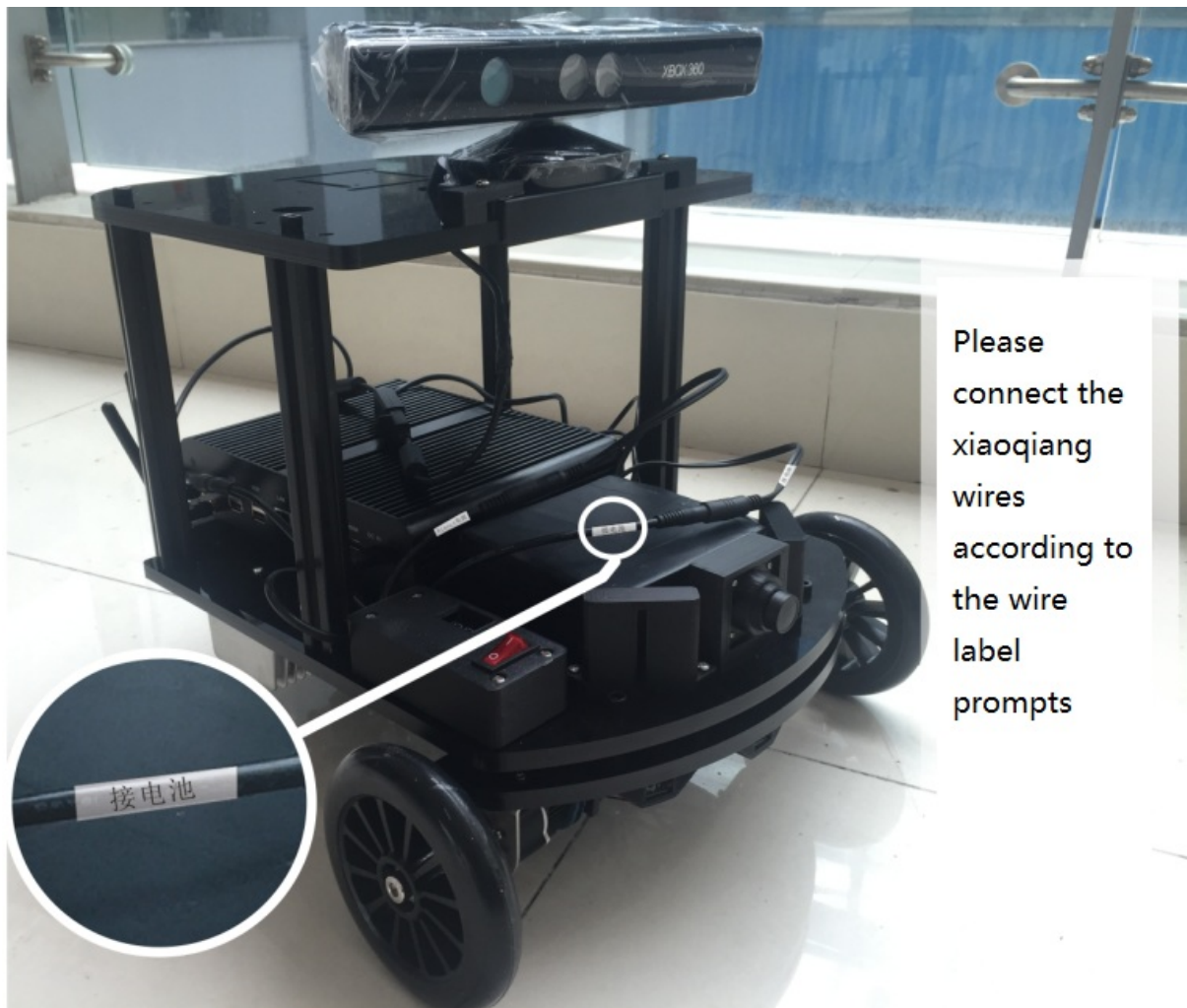
The following tutorial focuses on the use of various features of Xiaoqiang and does not include the teaching of ROS. If you are a beginner of ROS, follow up on the ROS basics while following the tutorials. Please don't just follow our tutorial and enter instructions. Understand the specific meaning of each command please. When you have problems, you should also be good at finding reasons by yourself, which will be helpful for future work and study. If you haven't understood about tutorials or ROS, please feel free to ask questions in [slack](#).

[Xiaoqiang HomePage](#)

[Buy Xiaoqiang](#)

[Buy Xiaoqiang mini](#)

First of all, please connect the xiaoqiang wires according to the wire label prompts. The complete structure is shown in the figure below.

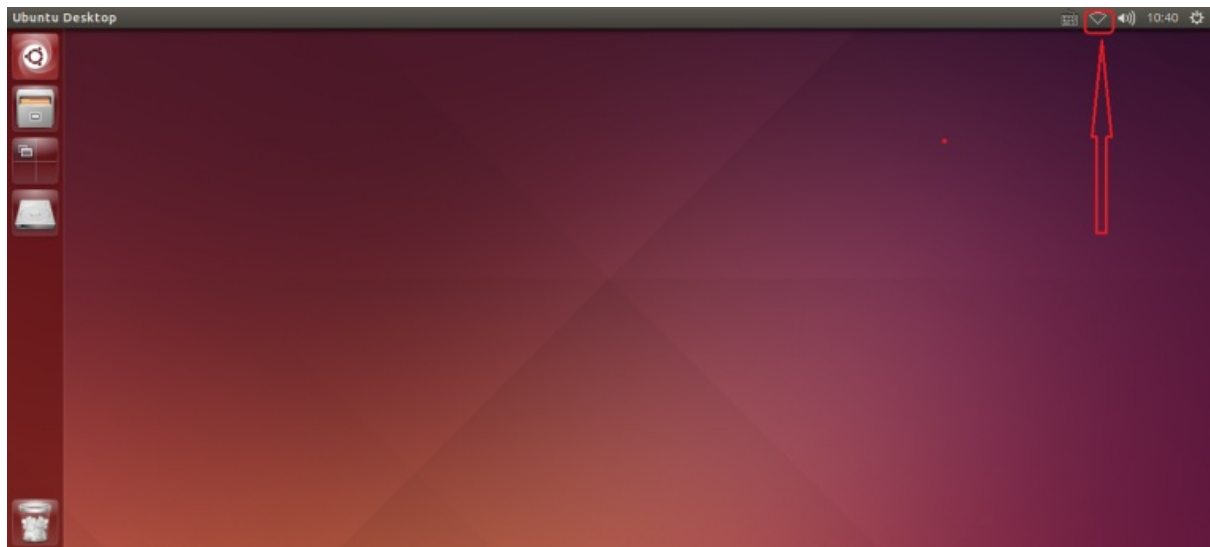


1. Configure the Xiaoqiang network

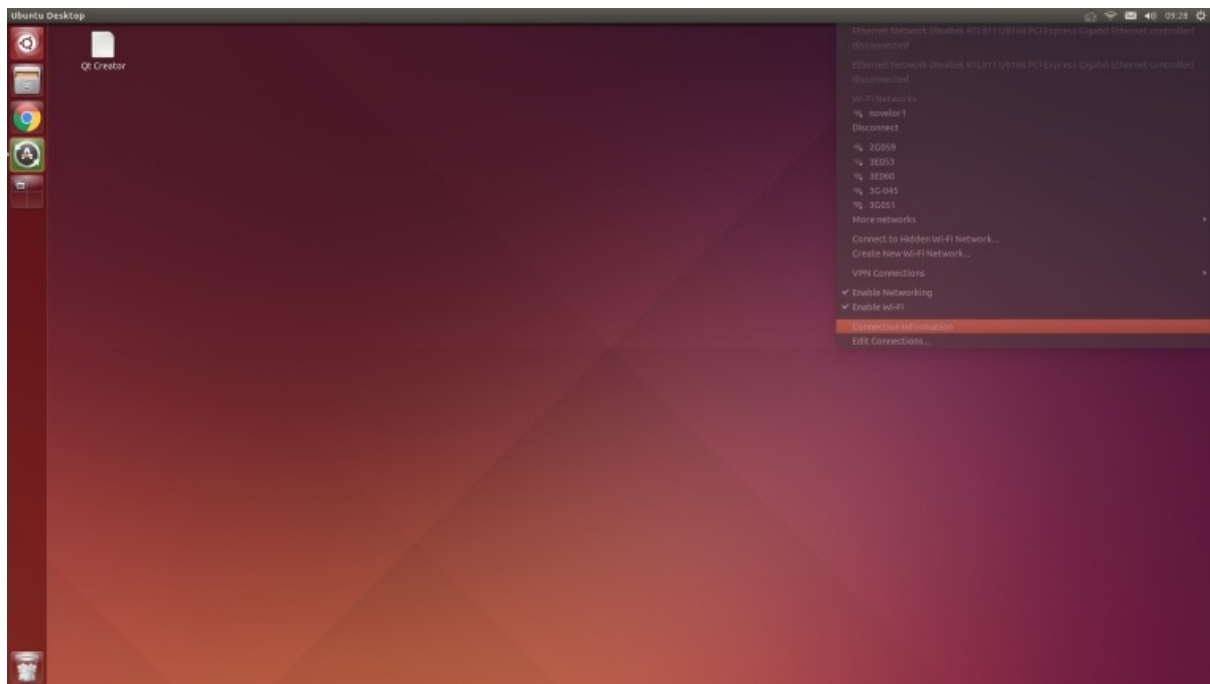
First connect Xiaoqiang's host computer with monitor (Xiaoqiang Pro users please use the included HDMI to VGA adapter to connect, Xiaoqiang Mini users please use vga directly) , The mouse and keyboard connected to the host computer. Then turn on the host computer and enter the Ubuntu system.

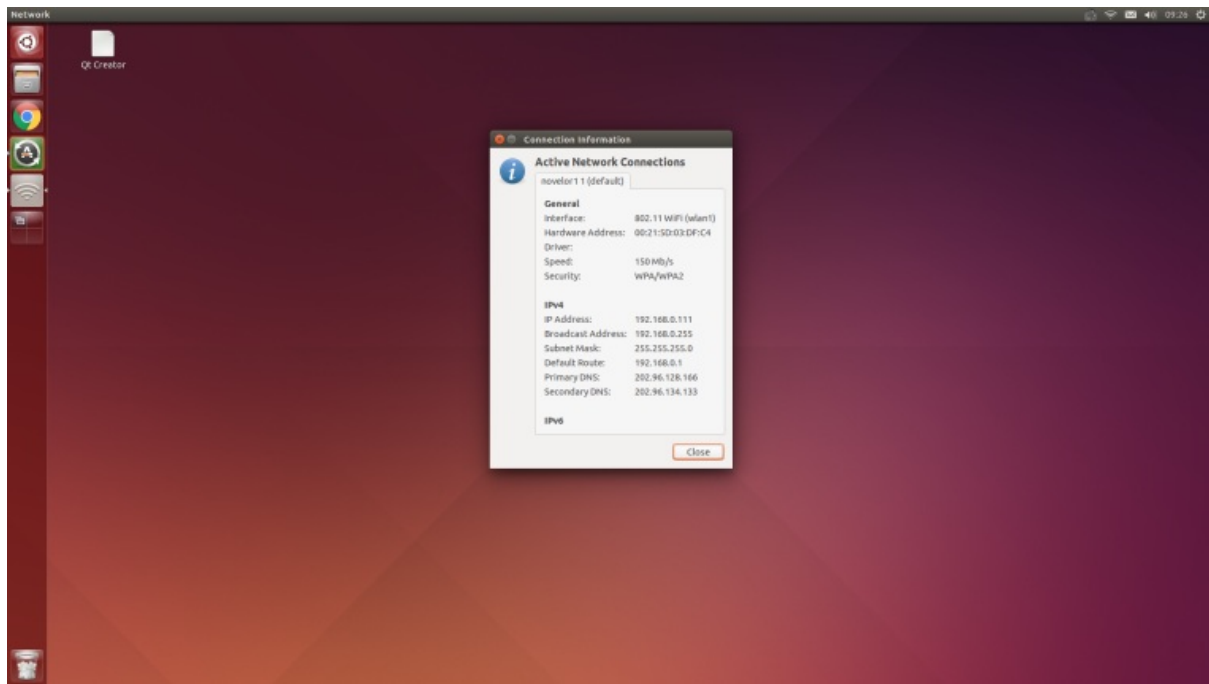
xiaoqiang's default password is xiaoqiang

Click on the location of the icon below and select the desired wireless network

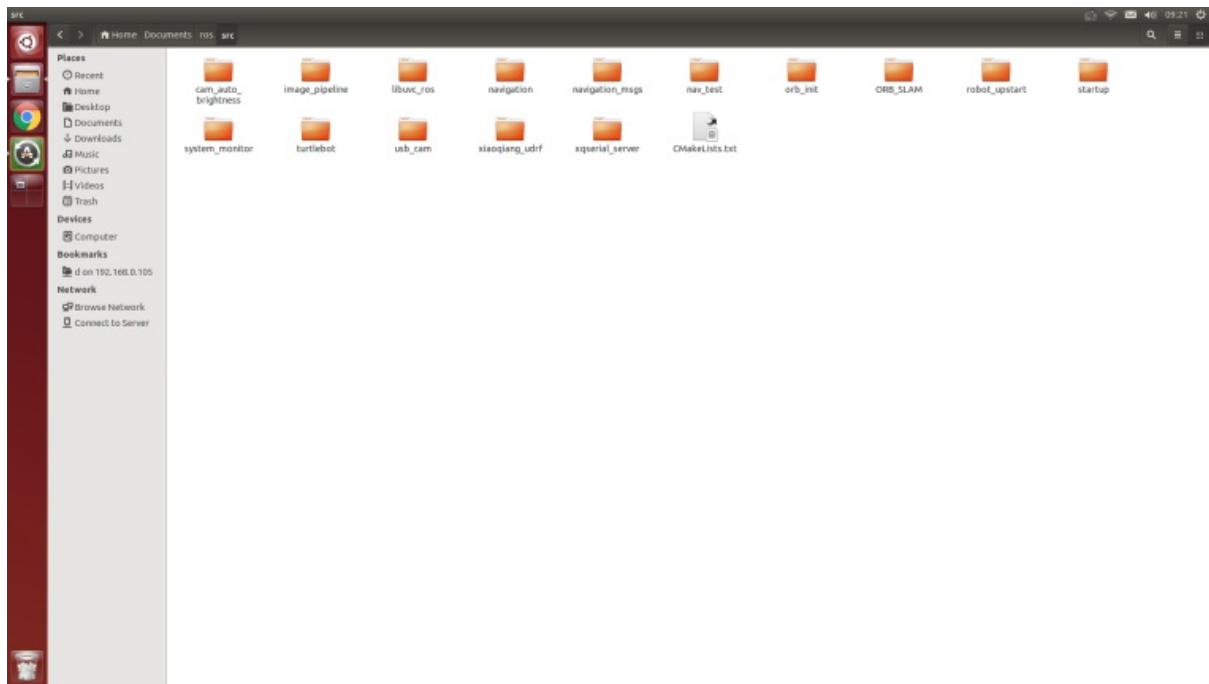


After accessing the network, click on the marked position in the figure below to get Xiaoqiang's actual ip address. Follow-up tutorials will use this ip information frequently.





Xiaoqiang's ROS default working directory is here



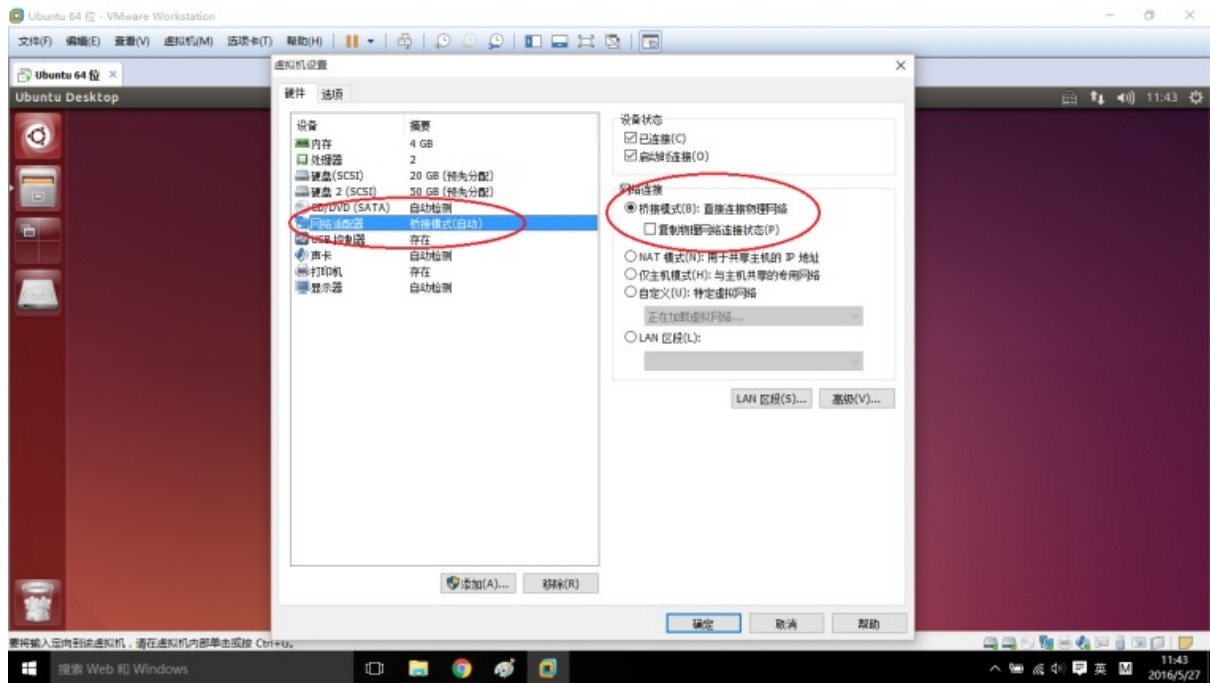
Now that Xiaoqiang has been configured, the keyboard, mouse, and monitor can be unplugged after power off. The next time Xiaoqiang is powered on, it can be used directly.

2. Local configuration

The local remote control end is preferably an x86 host with ubuntu 16.04 64 bit operating system installed. If you only have a windows platform, you can install the vmware virtual machine. Download Vmware Player from [here](#) , and then install ubuntu in the virtual machine. [Mirror download and installation tutorial](#), it is

recommended to install Xiaoqiang iso image(already configured ROS). Finally, ensure that the virtual machine be on the same LAN as Xiaoqiang.

Note The virtual machine NIC is to be set as a bridge, as shown in the following figure. Otherwise the virtual machine will not be able to link the xiaoqiang. If you are unable to connect to xiaoqiang, you can turn off your computer's firewall and try again.



Start Ubuntu system configuration after system deployment

a.Install `ssh`, `screen`, if you install Xiaoqiang mirror please skip this operation

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

b.Install ros kinetic, if you install Xiaoqiang mirror please skip this operation, [the official tutorial](#)

c.Users who installed Xiaoqiang system image, please turn off startup service to avoid conflicts with Xiaoqiang

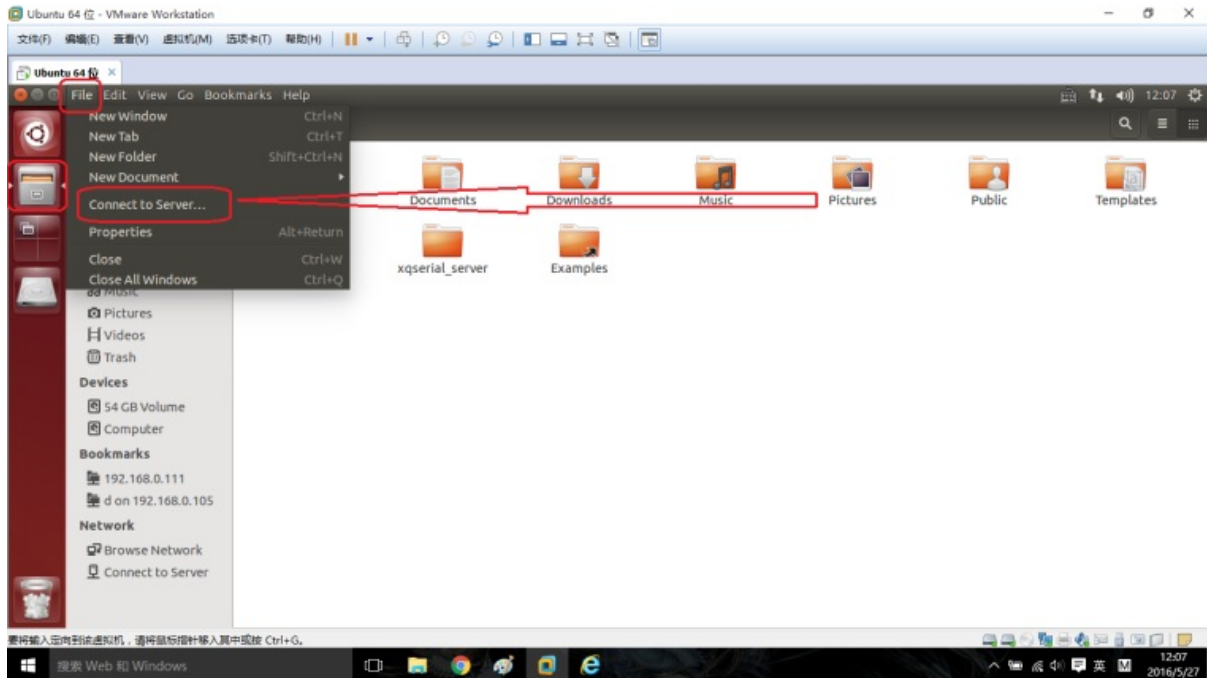
```
sudo service startup stop
rosrun robot_upstart uninstall startup
```

d.Users who installed Xiaoqiang mirror, please update Xiaoqiang_udrf package according to Xiaoqiang model(Pro?mini)

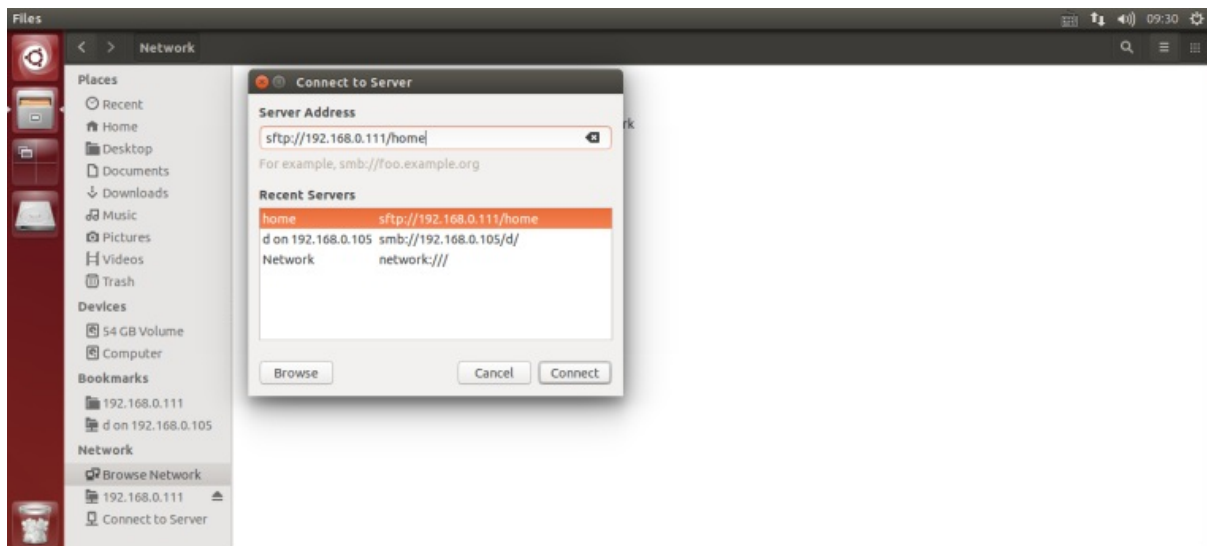
```
cd ~/Documents/ros/src/xiaoqiang_udrf
git stash
git pull
# Xiaoqiang pro user, please switch to the master branch
git checkout master
# Xiaoqiang mini user, please switch to mini branch
git checkout mini
```

e. Because the follow-up needs to frequently change the files on the Xiaoqiang host, we will now add the Xiaoqiang host remote directory to the local computer so that the file on the Xiaoqiang host can be graphically operated locally (the Xiaoqiang host is equivalent to the external hard disk of the local ubuntu system).

Click on the icon in the figure below to add the remote directory

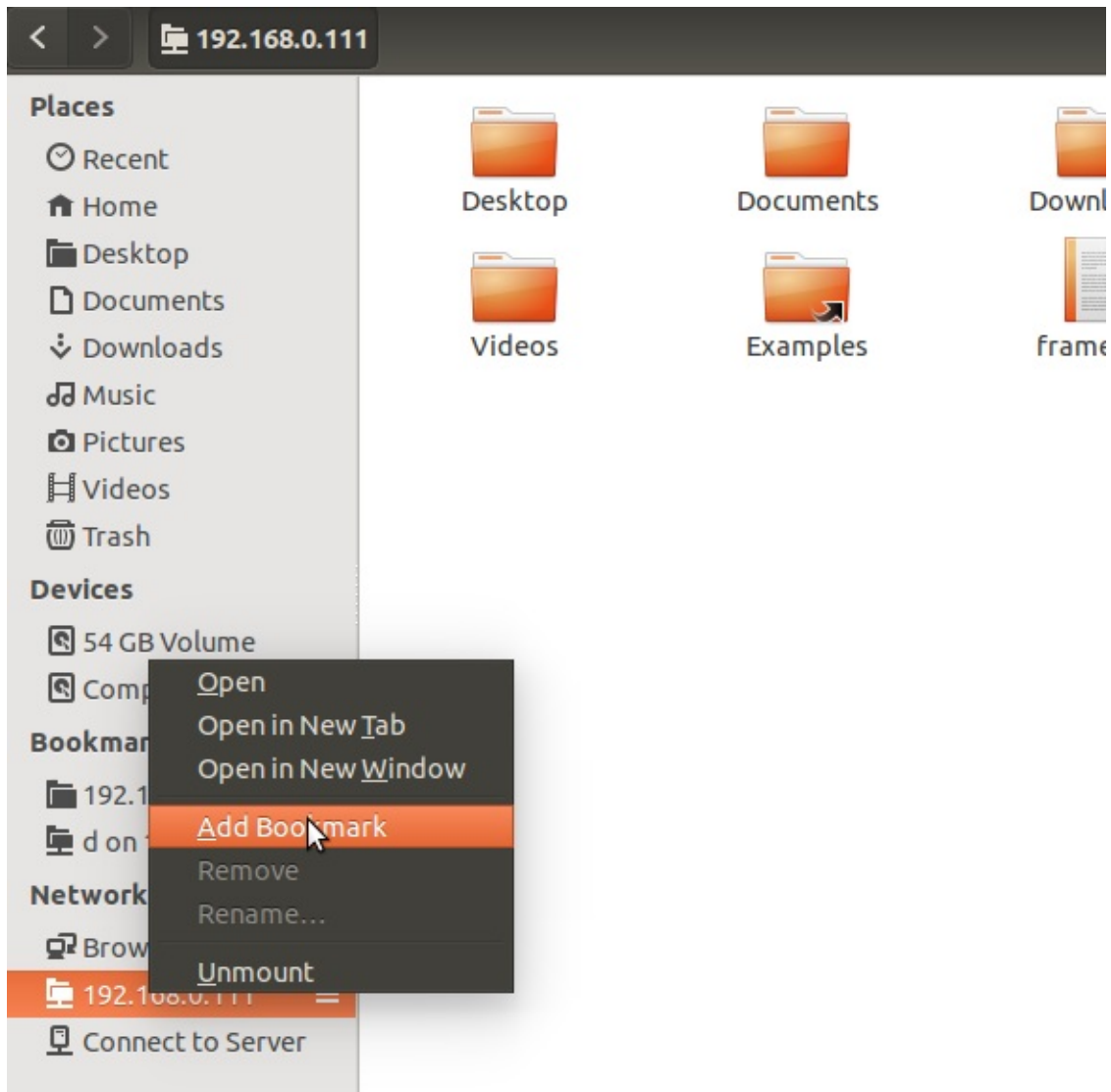


Enter Xiaoqiang's remote directory. Replace ip with the actual ip address mentioned above.

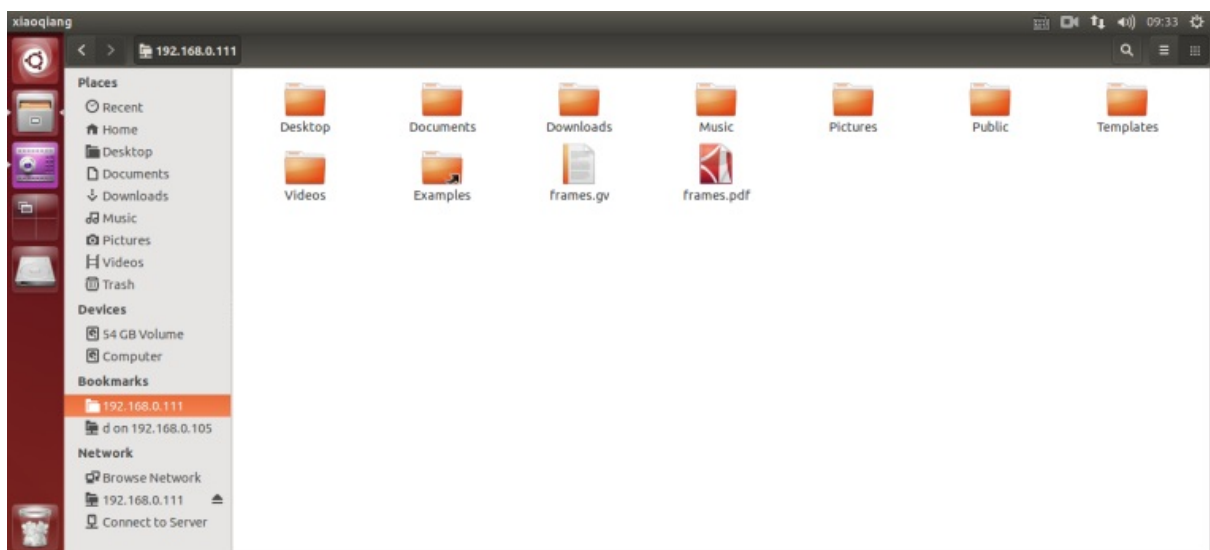


Enter the Xiaoqiang host username and password as prompted.

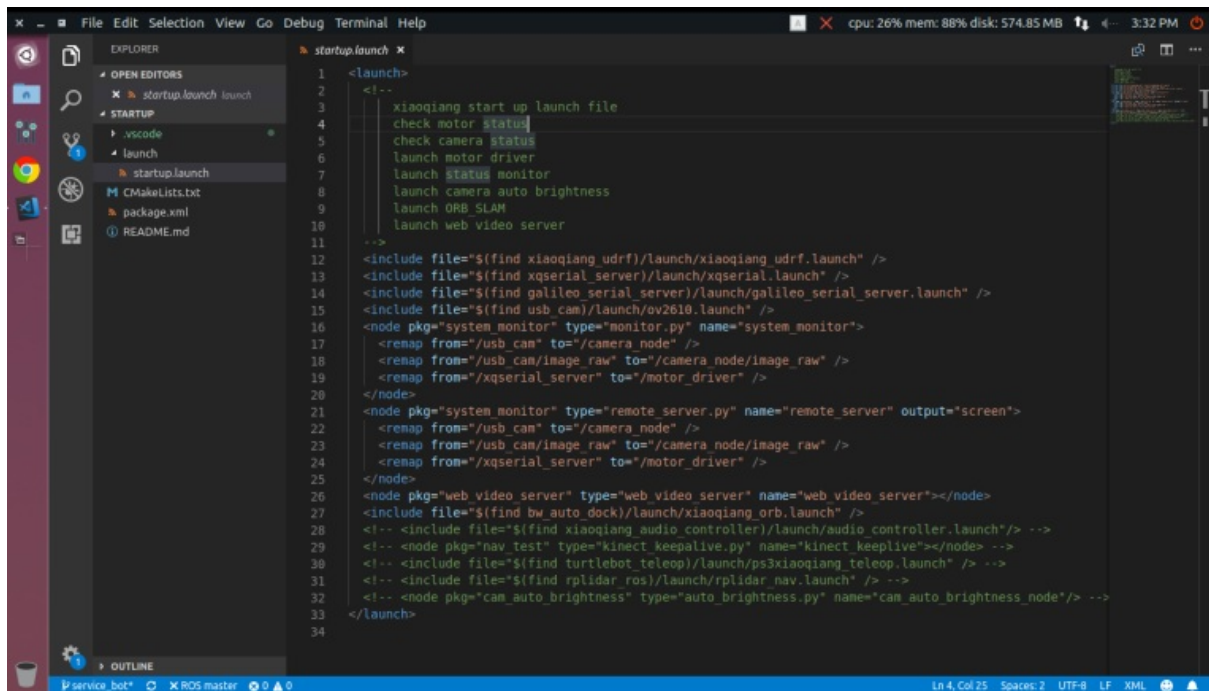
If everything is normal, the home directory of the Xiaoqiang host has been opened. For future convenience, this address can be added to the bookmark.



You click on this bookmark directly next time, you will be able to access the home directory of the Xiaoqiang host.



It is recommended to install the vscode editor (already installed in xiaoqiang system image), and it is convenient to code and edit the software package on the Xiaoqiang host.



e. Now that you have the full development and use environment of Xiaoqiang, for example, ssh logs in to the Xiaoqiang host and begins to use the keyboard to remotely control Xiaoqiang's movement.

1. Open a terminal on the local machine
2. Connect to xiaoqiang via ssh, replace xxx with actual ip

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

3. Start remote program

```
roslaunch nav_test control.py
```

4. You can start controlling the movement of Xiaoqiang through the arrow keys. Spacebar is the stop button. Press **Ctrl + C** to exits the program.

Xiaoqiang's mobile control is achieved by issuing a topic named `/cmd_vel`, This topic is the Twist type under `geometry_msgs`, Its release method please refer to ros official document. The control.py source code on Xiaoqiang is also a good reference example.

FAQ

Q: The car can only go forward and not back, or can only move back.

A: Check whether the car's infrared sensor is triggered and the sensor will light up after triggering. After the infrared trigger, the car cannot move in the corresponding direction.

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(2\) Bluewhale Robot Open source code repository usage and ROS startup task configuration](#)
- [Bluewhale Robot Open source code repository usage and ROS startup task configuration.](#)
 - [1. Startup package introduction](#)
 - [2. Download and install startup package on Xiaoqiang host](#)
 - [3. Modify the startup.launch file in the launch folder in the package](#)
 - [4. Register the startup.launch file as a startup service on Xiaoqiang host](#)
 - [5. Remotely restart Xiaoqiang's host and check if the boot startup item is loaded normally](#)

xiaoqiang tutorial (2) Bluewhale Robot Open source code repository usage and ROS startup task configuration

[Xiaoqiang Homepage](#)

Bluewhale Robot Open source code repository usage and ROS startup task configuration.

Some of Xiaoqiang's software source code is shared with Bluewhale Robot's [open source code repository](#), Anybody can freely download or use it for further development.

For Xiaoqiang users, the software in the open source Code repository can be directly git cloned into Xiaoqiang's ROS working directory, and then can be compiled and used directly with ROS's `catkin_make`. Xiaoqiang's ROS working directory is: `/home/xiaoqiang/Documents/ros/src`

The following will use the startup package in the open source repository as an example to demonstrate the complete use of the open source repository

1. Startup package introduction

After the Xiaoqiang host starts up, it will automatically start the Linux service script named startup. When the service script runs, it will start the replica which is registered in ubuntu system by `startup.launch` file. Therefore, by modifying the `startup.launch` file in the startup package, and then registering this file as a startup service in the ubuntu system, we can control the booting tasks of the Xiaoqiang host.

2. Download and install startup package on Xiaoqiang host

a. In the local remote control terminal ssh connection Xiaoqiang host, refer to the configuration of the previous tutorial

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

b. Enter Xiaoqiang's ROS working directory to see if there is a startup folder

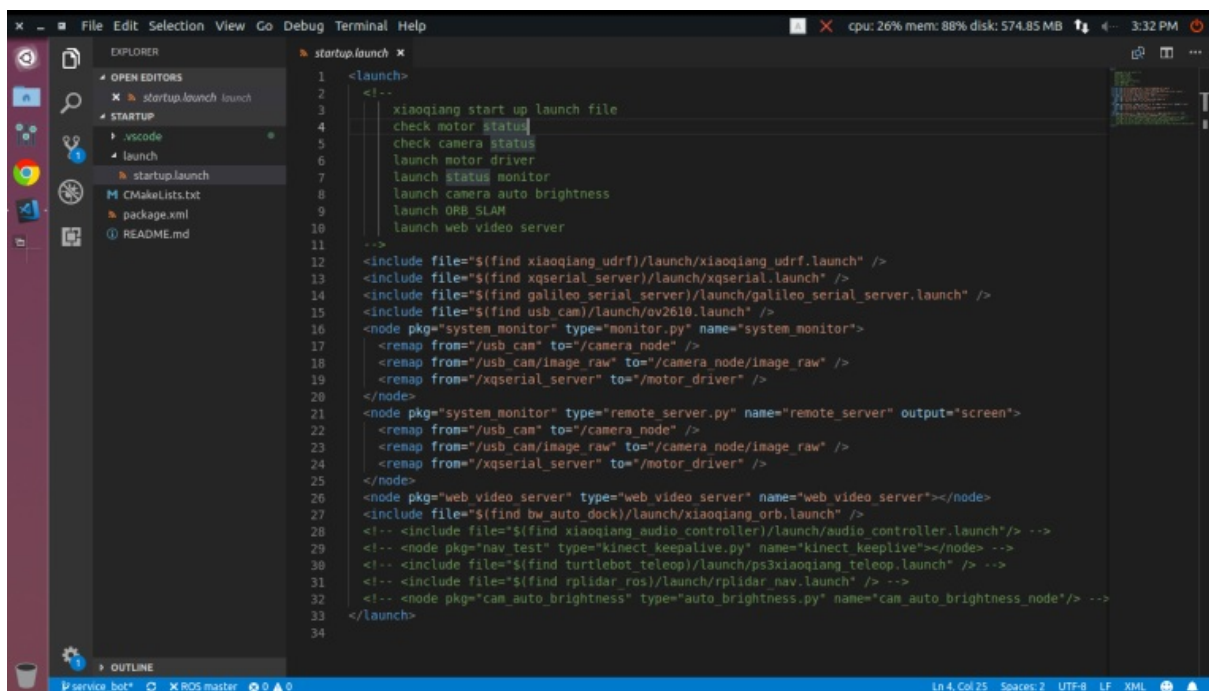
```
cd Documents/ros/src/
ls
```

If it exists, it indicates that the startup package has been installed. You can directly perform the following third step. If you want to update the startup package synchronously with the open source repository, enter the following command:

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

3.Modify the startup.launch file in the launch folder in the package

Using the vscode editor installed in the previous tutorial, Edit this file directly on the local machine (If you need to remotely access Xiaoqiang's host file directory, please refer to the previous basic operation tutorial to configure.)



```

1 <launch>
2 <!--
3   xiaoqiang start up launch file
4   check motor status
5   check camera status
6   launch motor driver
7   launch status monitor
8   launch camera auto brightness
9   launch ORB SLAM
10  launch web video server
11 -->
12 <include file="$(find xiaoqiang udrf)/launch/xiaoqiang_u drf.launch" />
13 <include file="$(find xqserial_server)/launch/xqserial.launch" />
14 <include file="$(find galileo_serial_server)/launch/galileo_serial_server.launch" />
15 <include file="$(find usb_cam)/launch/ov2610.launch" />
16 <node pkg="system_monitor" type="monitor.py" name="system_monitor">
17   <remap from="/usb_cam" to="/camera_node" />
18   <remap from="/usb_cam/image_raw" to="/camera_node/image_raw" />
19   <remap from="/xqserial_server" to="/motor_driver" />
20 </node>
21 <node pkg="system_monitor" type="remote_server.py" name="remote_server" output="screen">
22   <remap from="/usb_cam" to="/camera_node" />
23   <remap from="/usb_cam/image_raw" to="/camera_node/image_raw" />
24   <remap from="/xqserial_server" to="/motor_driver" />
25 </node>
26 <node pkg="web_video_server" type="web_video_server" name="web_video_server"></node>
27 <include file="$(find bw_auto_dock)/launch/xiaoqiang_orb.launch" />
28 <!-- <include file="$(find xiaoqiang_audio_controller)/launch/audio_controller.launch"/> -->
29 <!-- <node pkg="nav_test" type="kinect_keeplive.py" name="kinect_keeplive"></node> -->
30 <!-- <include file="$(find turtlebot_teleop)/launch/ps3xiaoqiang_teleop.launch" /> -->
31 <!-- <include file="$(find rplidar_ros)/launch/rplidar_nav.launch" /> -->
32 <!-- <node pkg="cam_auto_brightness" type="auto_brightness.py" name="cam_auto_brightness_node"/> -->
33 </launch>
34

```

In the arrow area above, add or delete the ROS launch file and ROS node that you need to launch. These items will be added to the boot of the Xiaoqiang host in the following. Xiaoqiang will automatically run these items when booted next time. Finally, save and exit.

4.Register the startup.launch file as a startup startup service on Xiaoqiang host

Please continue to enter in the ssh window in the second section above.

- a. First stop and delete the previously registered startup service

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

- b. Re-register startup service

```
rosrun robot_upstart install startup/launch/startup.launch
sudo systemctl daemon-reload && sudo systemctl start startup
```

5. Remotely restart Xiaoqiang's host and check if the boot startup item is loaded normally

Then enter the ssh window above

- a. Send a reboot command

```
sudo shutdown -r now
```

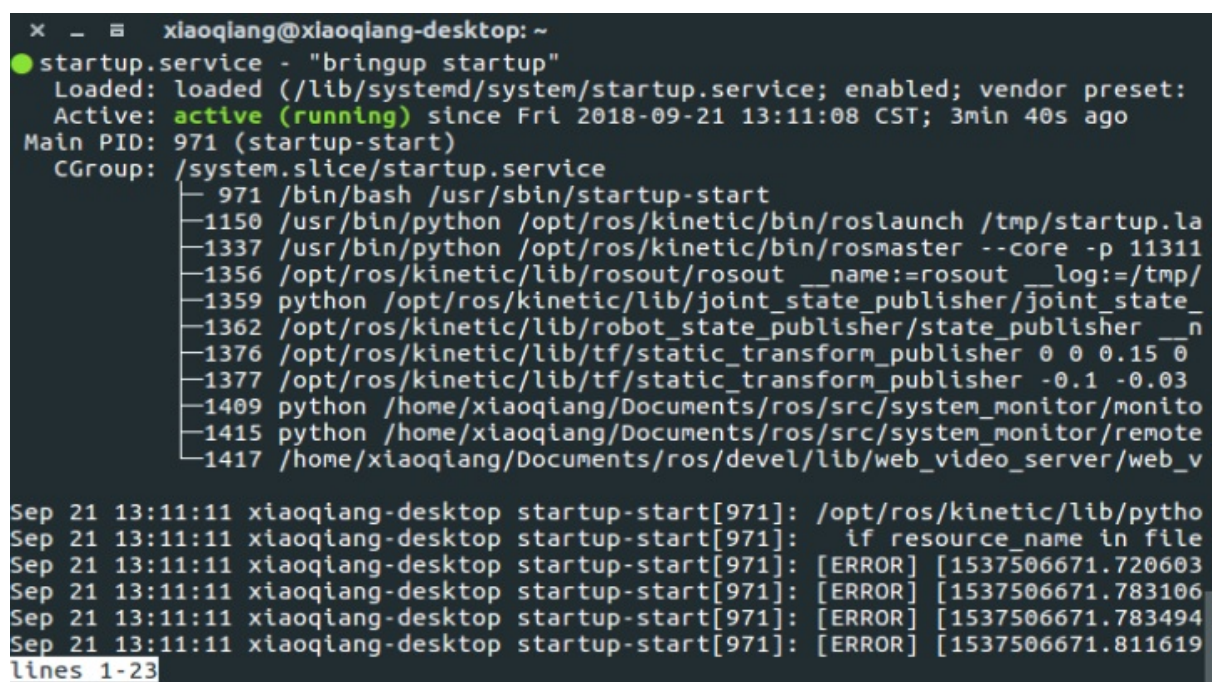
- b. Start a new ssh connection

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

- c. Check the status of the startup service

```
sudo service startup status
```

Normally, it will show startup start/running as shown in the figure below.



```
xiaoqiang@xiaoqiang-desktop: ~
● startup.service - "bringup startup"
   Loaded: loaded (/lib/systemd/system/startup.service; enabled; vendor preset:
   Active: active (running) since Fri 2018-09-21 13:11:08 CST; 3min 40s ago
   Main PID: 971 (startup-start)
   CGroup: /system.slice/startup.service
           └─ 971 /bin/bash /usr/sbin/startup-start
              └─ 1150 /usr/bin/python /opt/ros/kinetic/bin/roslaunch /tmp/startup.la
                 └─ 1337 /usr/bin/python /opt/ros/kinetic/bin/rosmaster --core -p 11311
                    └─ 1356 /opt/ros/kinetic/lib/rosout/rosout __name:=rosout __log:=/tmp/
                       └─ 1359 python /opt/ros/kinetic/lib/joint_state_publisher/joint_state_
                          └─ 1362 /opt/ros/kinetic/lib/robot_state_publisher/state_publisher __n
                             └─ 1376 /opt/ros/kinetic/lib/tf/static_transform_publisher 0 0 0.15 0
                                └─ 1377 /opt/ros/kinetic/lib/tf/static_transform_publisher -0.1 -0.03
                                   └─ 1409 python /home/xiaoqiang/Documents/ros/src/system_monitor/monito
                                      └─ 1415 python /home/xiaoqiang/Documents/ros/src/system_monitor/remote
                                         └─ 1417 /home/xiaoqiang/Documents/ros/devel/lib/web_video_server/web_v

Sep 21 13:11:11 xiaoqiang-desktop startup-start[971]: /opt/ros/kinetic/lib/pytho
Sep 21 13:11:11 xiaoqiang-desktop startup-start[971]:   if resource_name in file
Sep 21 13:11:11 xiaoqiang-desktop startup-start[971]: [ERROR] [1537506671.720603
Sep 21 13:11:11 xiaoqiang-desktop startup-start[971]: [ERROR] [1537506671.783106
Sep 21 13:11:11 xiaoqiang-desktop startup-start[971]: [ERROR] [1537506671.783494
Sep 21 13:11:11 xiaoqiang-desktop startup-start[971]: [ERROR] [1537506671.811619
lines 1-23
```

d. You can also further see if the related topic has been published

```
rostopic list
```

[Xiaoqiang Homepage](#) [Back to Index](#)

- [xiaoqiang tutorial \(3\) Displaying Xiaoqiang Robot Model in rviz](#)
 - [Displaying Xiaoqiang Robot Model in rviz](#)

xiaoqiang tutorial (3) Displaying Xiaoqiang Robot Model in rviz

[Xiaoqiang Homepage](#)

Displaying Xiaoqiang Robot Model in rviz

It is a cool thing to display the current pose of the robot in real time. With rviz, you can easily achieve this goal in ROS. Look at the results:



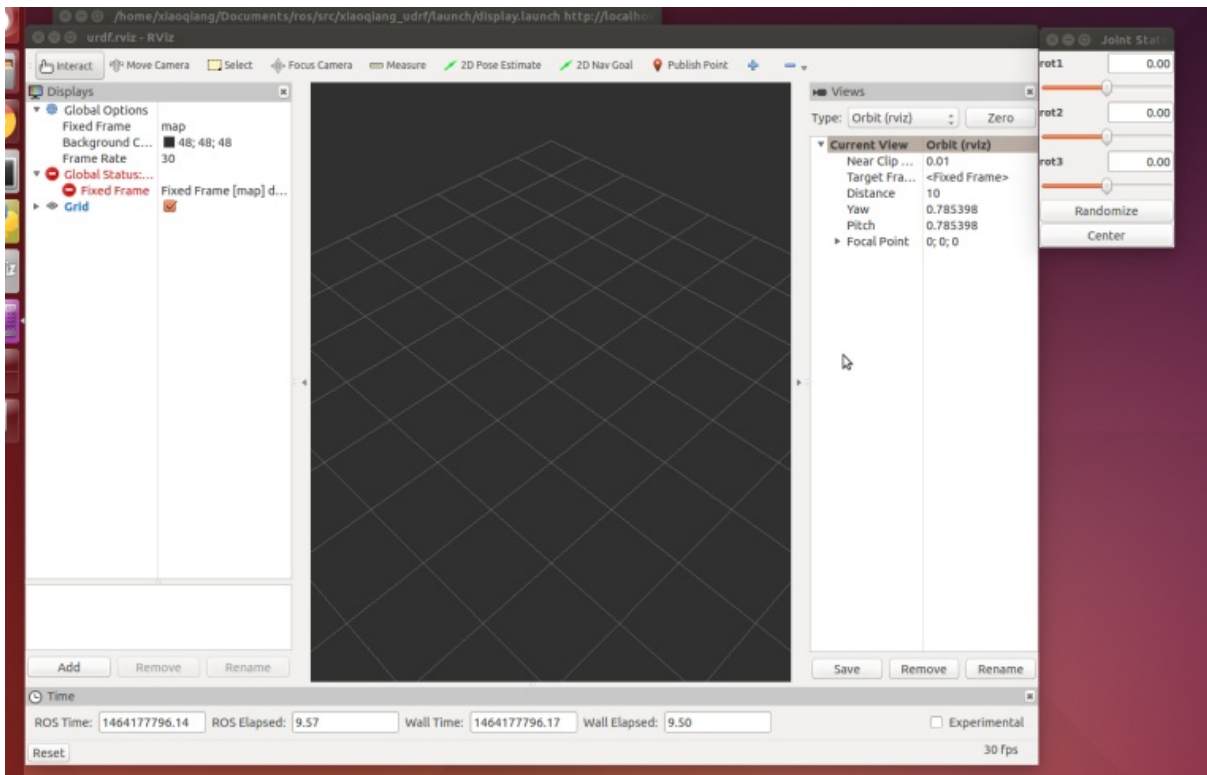
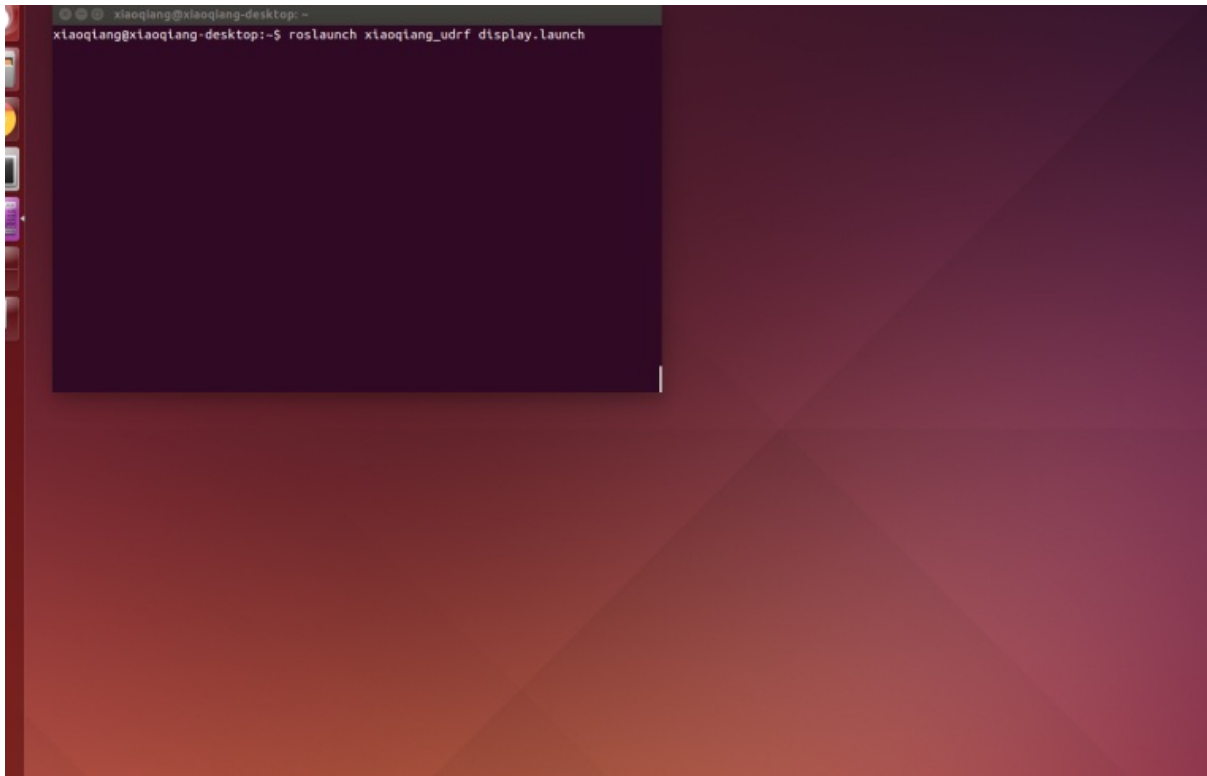
The Xiaoqiang model package located at `Documents/ros/src/xiaoqiang_udrf` and can also be downloaded from our open source repository on Github.

```
# Connect the Xiaoqiang host to the monitor and keyboard, and after booting, open the terminal
# stop the startup task first
sudo service startup stop
roscore
# Open a new terminal on Xiaoqiang host, start this package
roslaunch xiaoqiang_udrf display.launch
```

Note that this process requires a graphical interface, so there is no way to operate through a remote ssh connection. If you must use ssh connection, you can add `-X` (note the capital) option when connecting, but the experience is not good and is not recommended. It is recommended to operate through VNC. VNC installation method can refer to this [article](#).

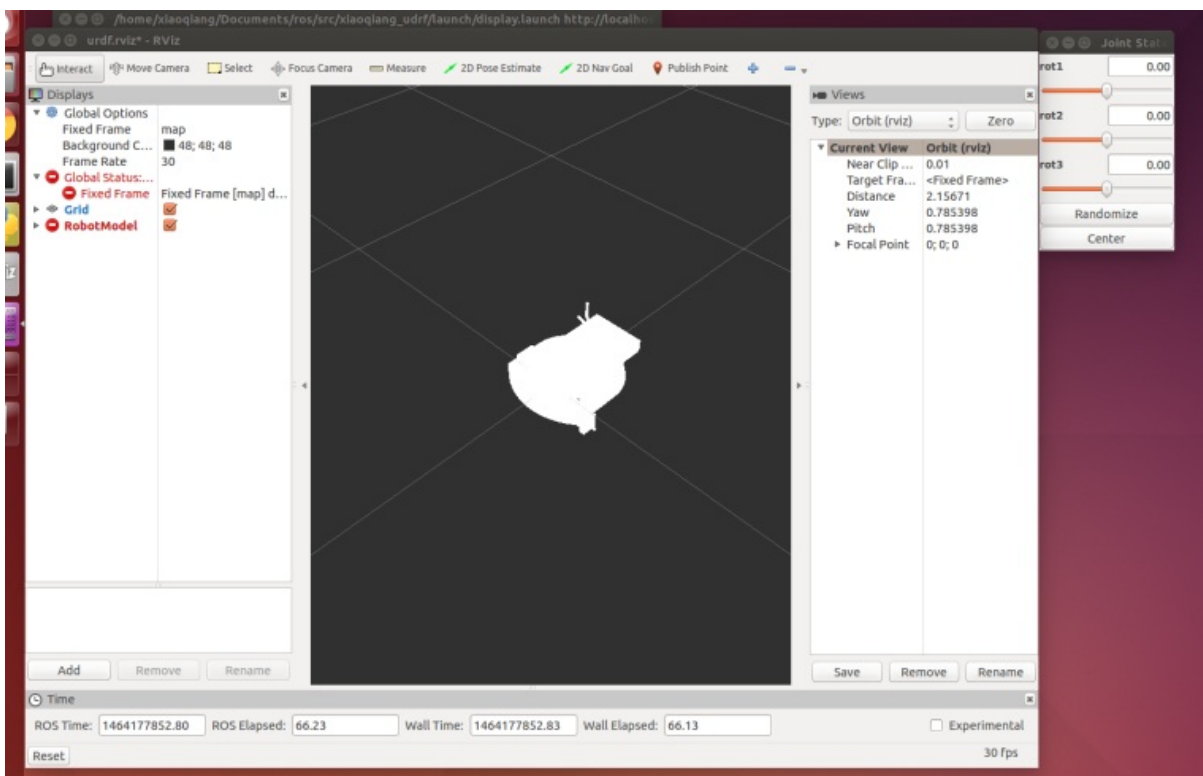
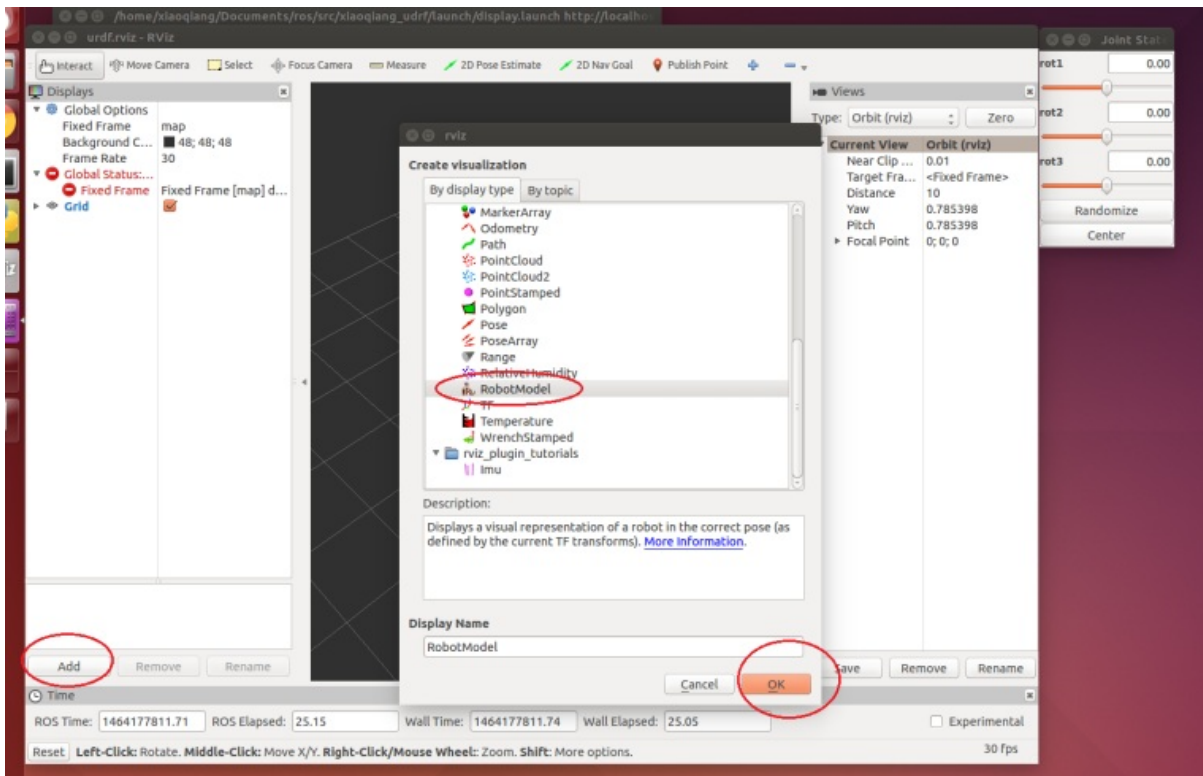
If you want to open the model package remotely, you need to install Xiaoqiang's model package locally. For users using the Xiaoqiang system image, Xiaoqiang's model package has been installed in advance.

3. Displaying Xiaoqiang Robot Model in rviz

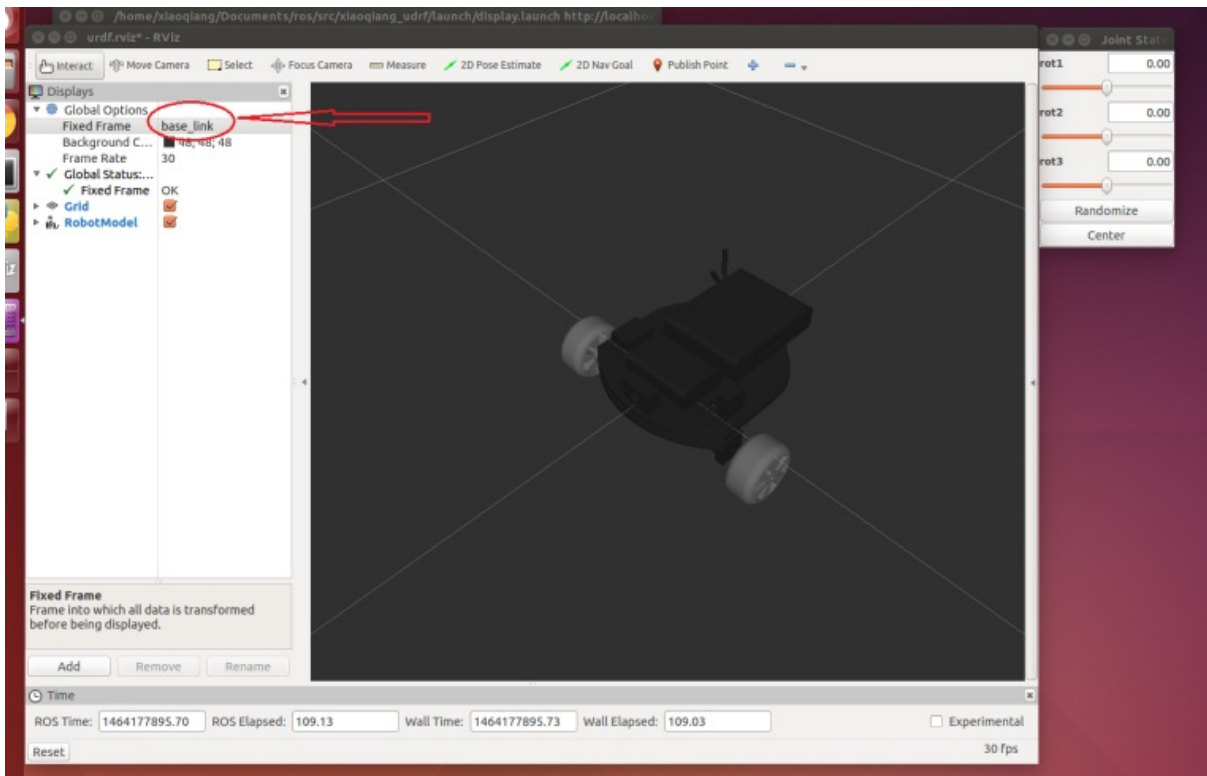


At this point, there is no thing to display, you need to add rviz display items

3. Displaying Xiaoqiang Robot Model in rviz



There is still a problem, the entire model is transparent and whitish, this is because the global coordinate system fixed frame in the rviz is not set properly, After map is changed to `base_link`, it will display normally

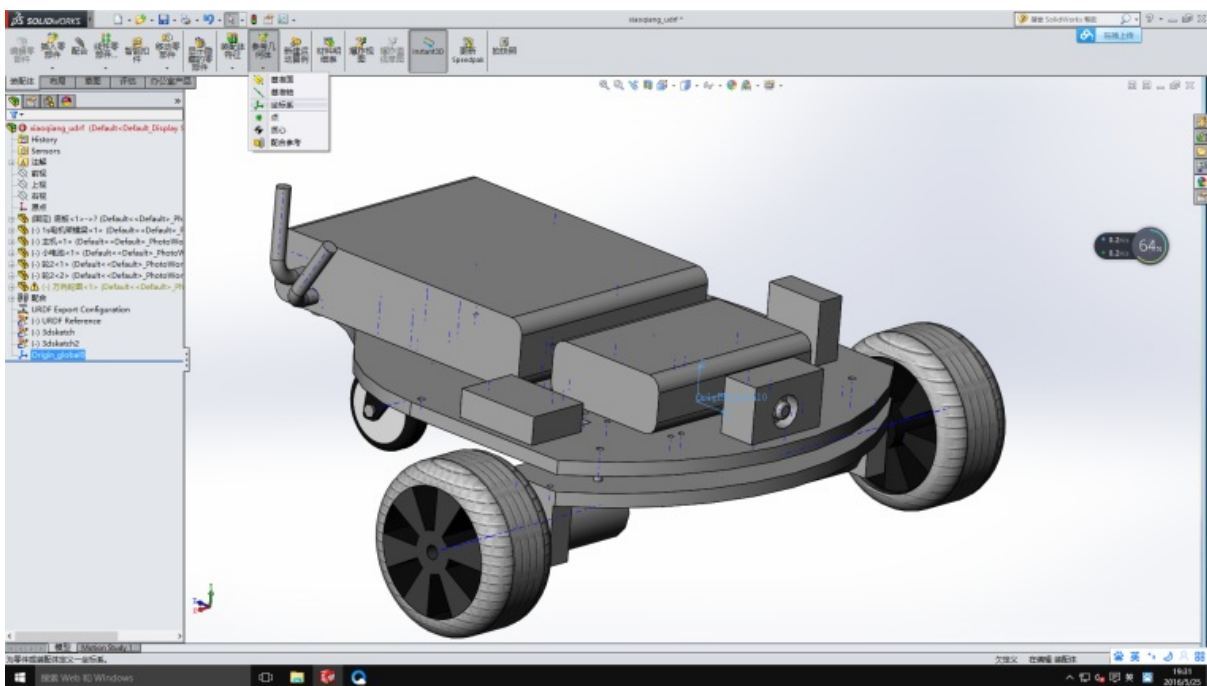


Now operate the slider in the upper right corner to turn the corresponding wheel.

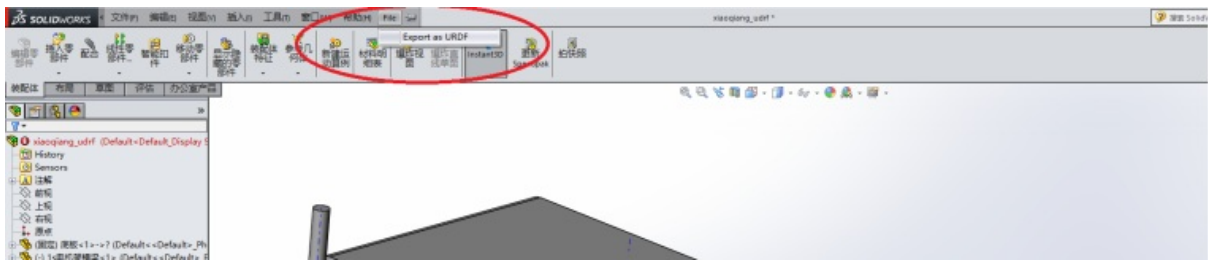
Above we simply demonstrated the use of rviz to display the urdf model. The following section will describe in detail the whole process of making urdf model with solidworks under windows system.

First use solidworks to create the platform model, download and install the solidwork to [urdf plug-in](#).

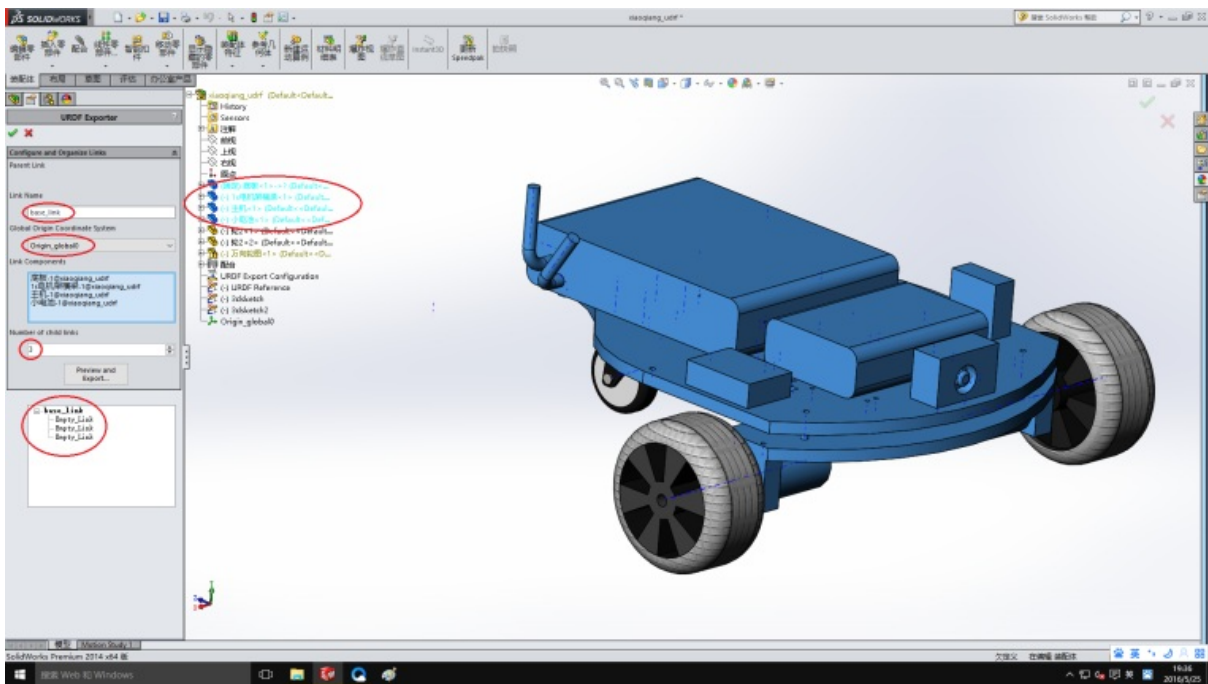
After making a model, you need to add a coordinate system. This coordinate system will be used as the reference coordinate system of the entire urdf model (ie, the base_link frame in ROS).



Open urdf plugin

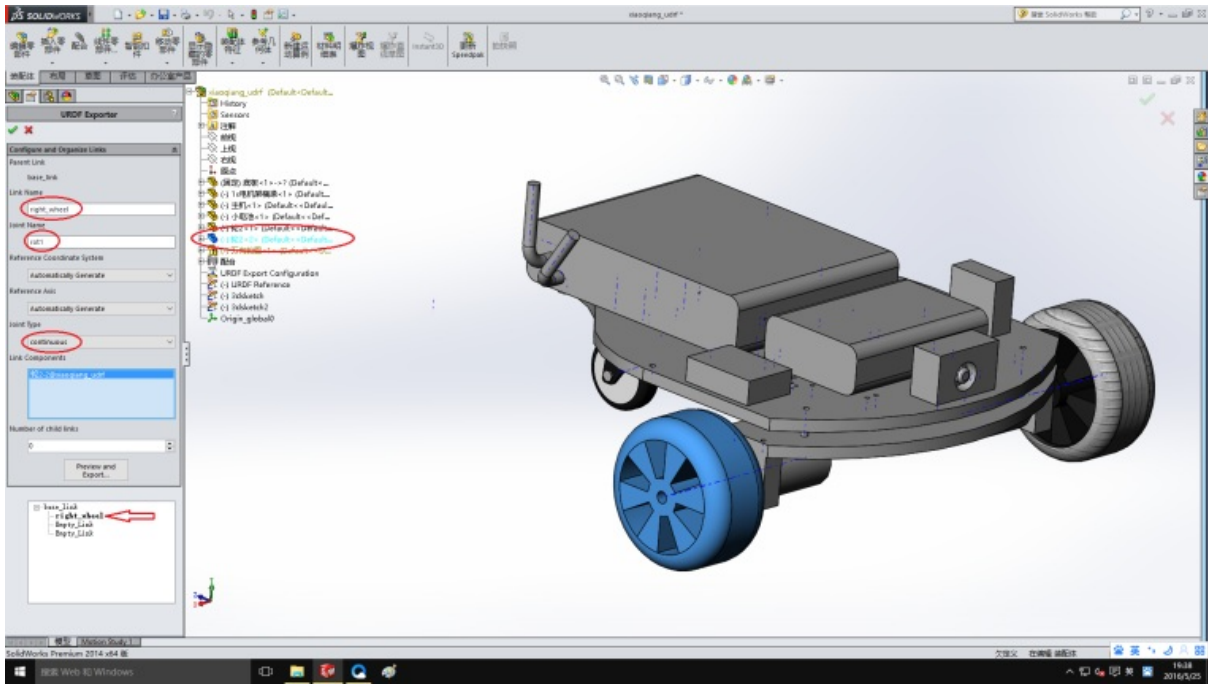


Xiaoqiang has two driving wheels and one driven wheel, so the whole model needs 3 links, 3 joints. First edit the `base_link`, pay attention to the global coordinate system above, the red area in the picture is the item you need to click or modify.

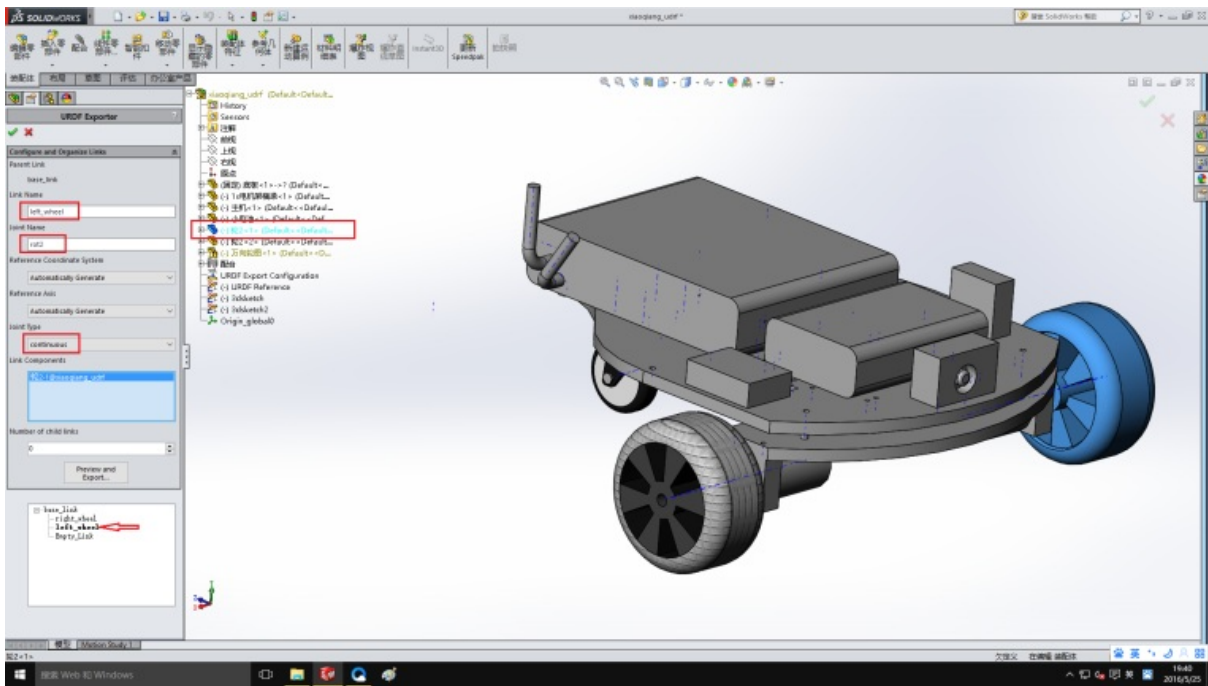


Then right wheel

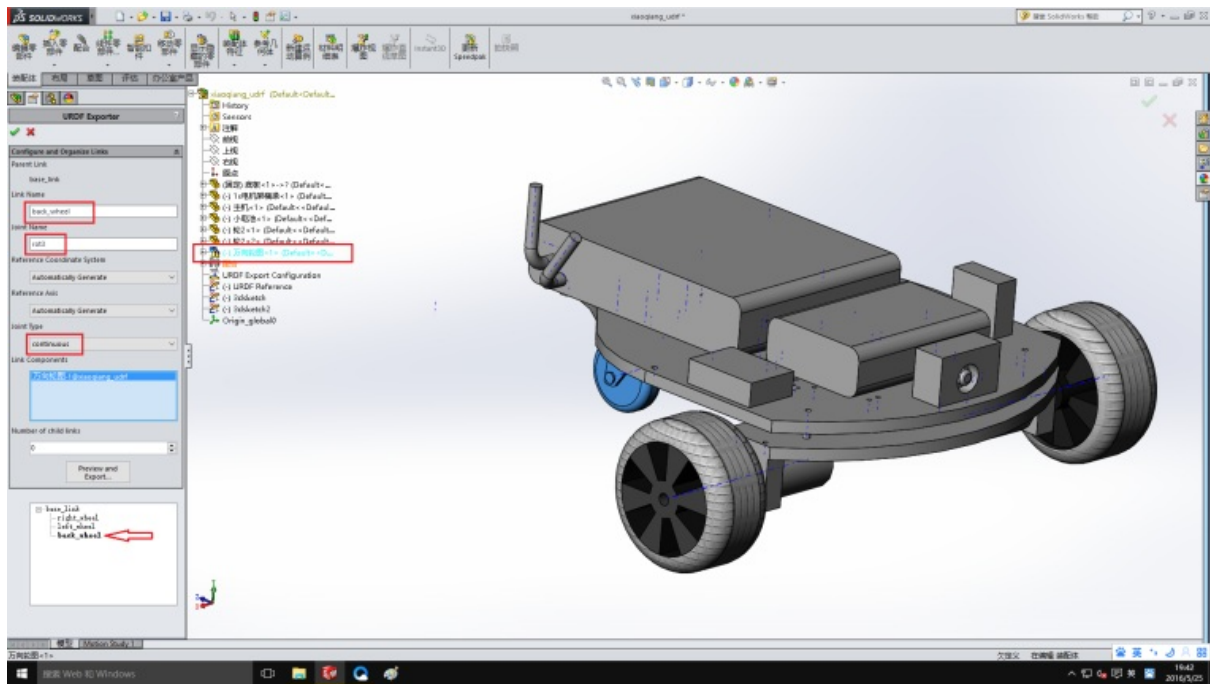
3. Displaying Xiaoqiang Robot Model in rviz



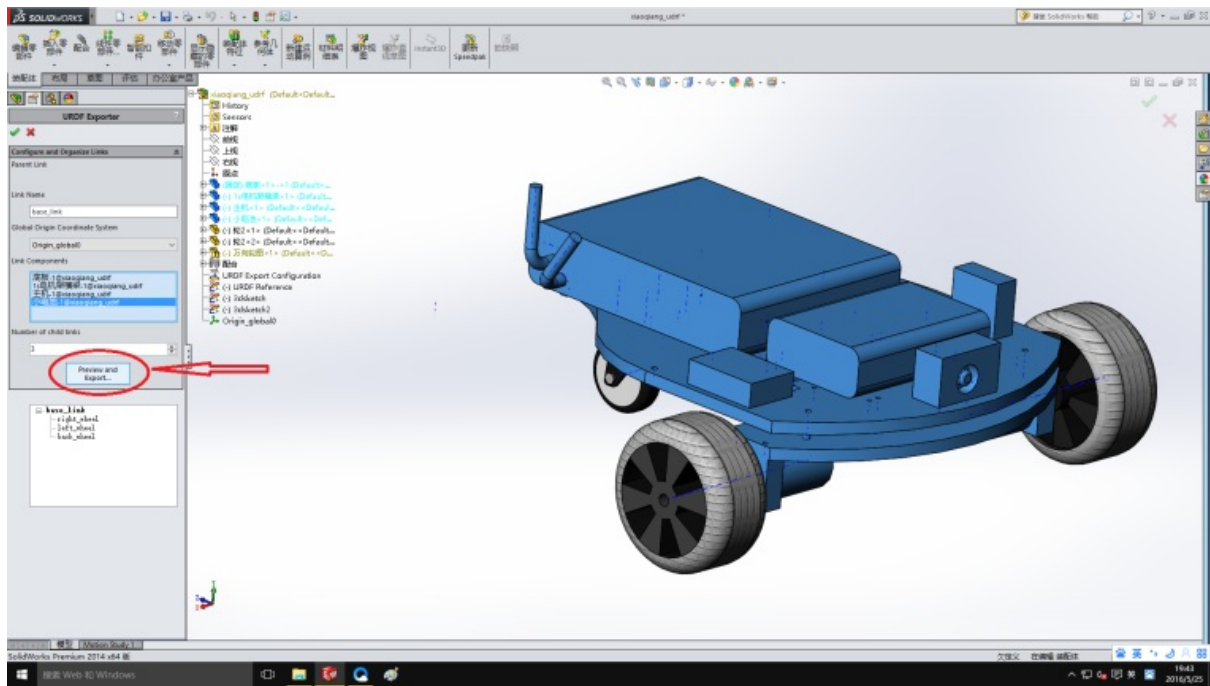
Continue the revolver and rear wheel



3. Displaying Xiaoqiang Robot Model in rviz

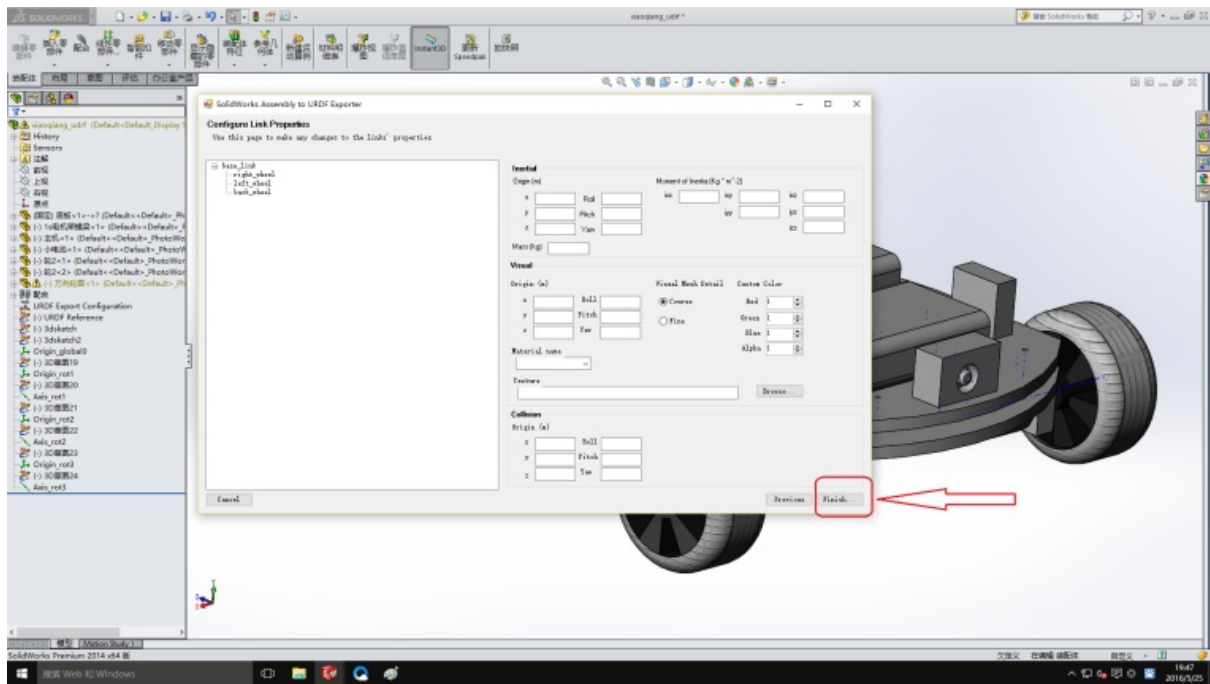
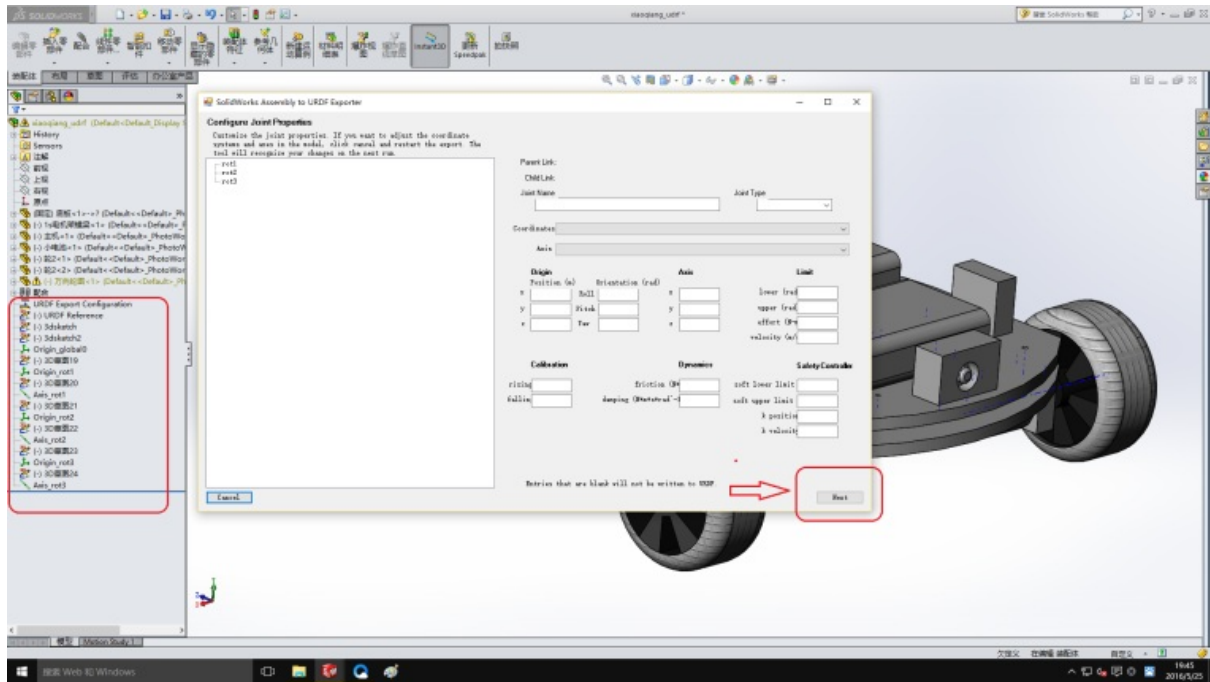


All settings are now complete, start exporting

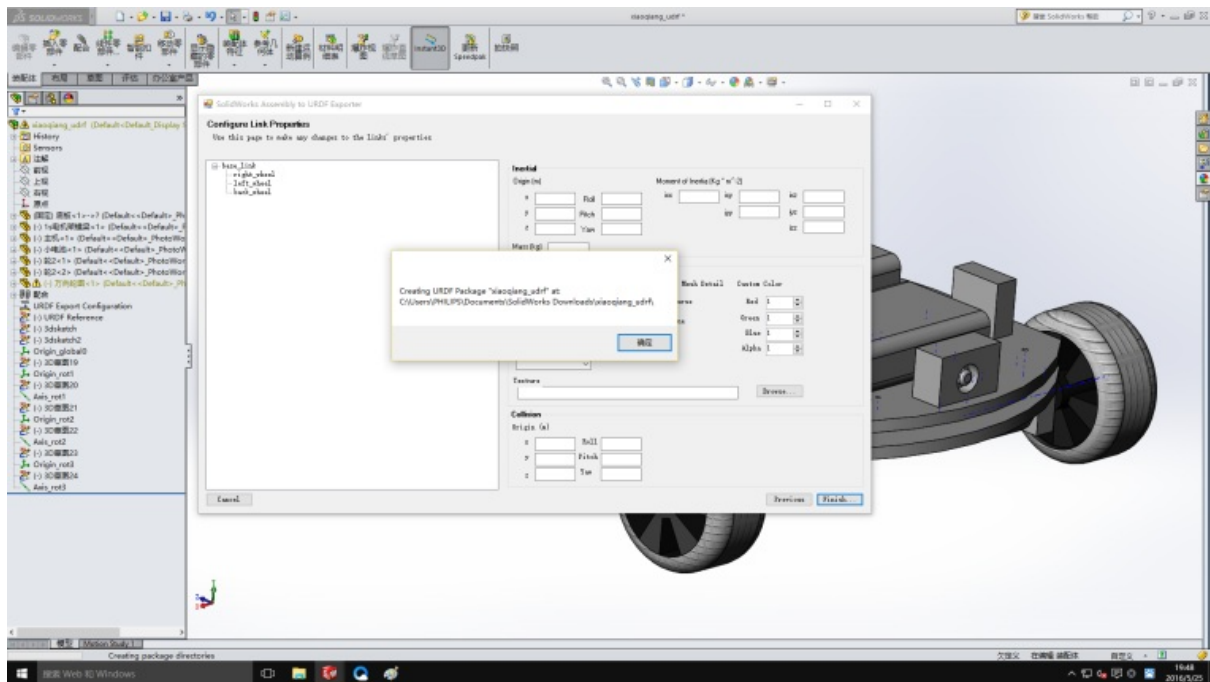


All the way next and ok go

3. Displaying Xiaoqiang Robot Model in rviz



3. Displaying Xiaoqiang Robot Model in rviz



Now that we have obtained the robot's urdf file, the entire generated folder is a ROS package. Modify the display.launch file in the launch folder and change false to true

```
display.launch
1 <launch>
2 <arg
3   name="model" />
4 <arg
5   name="gui"
6   default="True" />
7 <param
8   name="robot_description"
9   textfile="$(find xiaoqiang_urdf)/robots/xiaoqiang_urdf.URDF" />
10 <param
11   name="use_gui"
12   value="$(arg gui)" />
13 <node
14   name="joint_state_publisher"
15   pkg="joint_state_publisher"
16   type="joint_state_publisher" />
17 <node
18   name="robot_state_publisher"
19   pkg="robot_state_publisher"
20   type="state_publisher" />
21 <node
22   name="rviz"
23   pkg="rviz"
24   type="rviz"
25   args="-d $(find xiaoqiang_urdf)/urdf.rviz" />
26 </launch>
27
```

Copy this ros package to the ROS workspace. After running catkin_make, it can be tested and used by the method at the beginning of this article.

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(4\) Inertial navigation test](#)
 - [inertial navigation test](#)
 - 1. remote operation on Xiaoqiang host
 - 2. Operation completed on the local machine
 - 3. Complete the last operation in the remote host ssh window
 - 4. Now you can see the video effect at the beginning of the article in rviz, enjoy it!

xiaoqiang tutorial (4) Inertial navigation test

[Xiaoqiang Homepage](#)

inertial navigation test

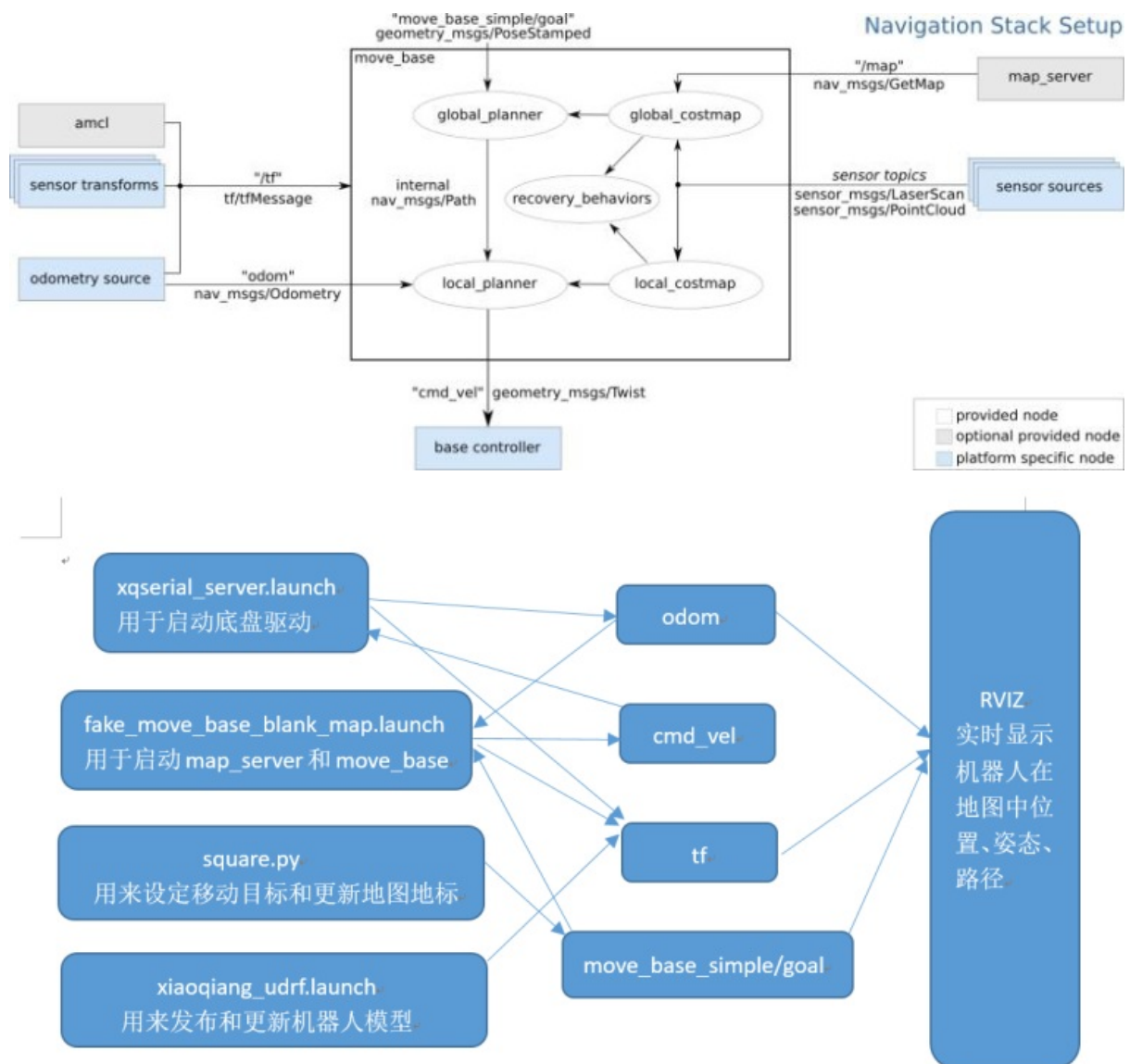
In the previous tutorial, we have made a 3D model of xiaoqiang, we will start to test xiaoqiang's inertial navigation function. The inertial navigation here is to use Xiaoqiang's own inertial sensors (acceleration and gyroscope) and encoder information to locate. The required ROS packages are:

1. Driver [xqserial_server](#)
2. Robot model package [xiaoqiang_udrf](#)
3. Inertial Navigation Test Package [nav_test](#).

Below is the final result of the tutorial.



the structure of ROS navigation stack is shown in the following two figures: If you are not familiar with the ROS navigation stack, please move to this [tutorial](#)



1. remote operation on Xiaoqiang host

Please ensure that Xiaoqiang has started normally. Xiaoqiang host will automatically run the three software packages mentioned after normal startup. You don't need to launch the corresponding launch file manually. If you don't have the three packages installed, please install and upgrade these three software packages yourself. Then update the startup task (please refer to the [robot_upstart tutorial](#)) and restart it.

a. Open a new terminal and start the navigation program

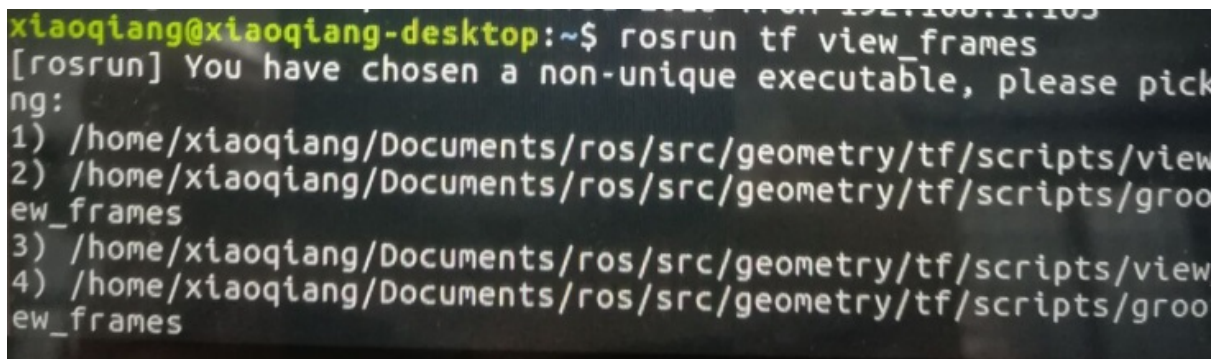
```
ssh -X xiaoqiang@192.168.xxx.xxx
# Restart the service program, some program may not be running
sudo service startup restart
roslaunch nav_test fake_move_base_blank_map.launch
```

If there is an error saying "You must specify at least three point for the robot footprint", Just ignore it. This is because `foot_print` requires setting a shape, so at least three points are required. `foot_print` is the outline shape of the robot. This parameter is used to avoid obstacles. Xiaoqiang's `foot_print` sets a circle, so there is no need to specify three points. If the tf tree that appears is not the same as the following figure, please check whether the drive is running normally.

b. Open a new terminal and check if all tf are in place

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

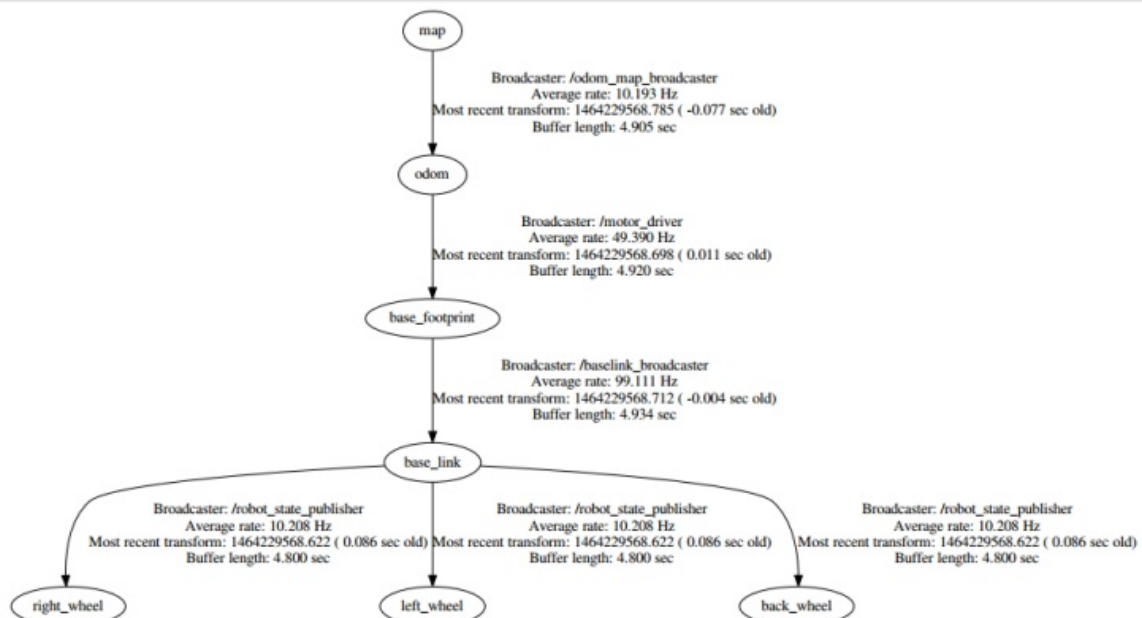
If the output looks like following



```
xiaoqiang@xiaoqiang-desktop:~$ roslaunch tf view_frames
[roslaunch] You have chosen a non-unique executable, please pick
ng:
1) /home/xiaoqiang/Documents/ros/src/geometry/tf/scripts/view
2) /home/xiaoqiang/Documents/ros/src/geometry/tf/scripts/groo
ew_frames
3) /home/xiaoqiang/Documents/ros/src/geometry/tf/scripts/view
4) /home/xiaoqiang/Documents/ros/src/geometry/tf/scripts/groo
ew_frames
```

Follow the prompts to select one `view_frame` .

Normal will show the following picture



2. Operation completed on the local machine

The local machine must be a Ubuntu system with corresponding ROS installed. Please refer to Section 1.2 of [Tutorial \(1\)](#) to install Xiaoqiang System Image. At the same time, it is guaranteed to be in the same local area network as Xiaoqiang. Because it is necessary to use rviz to display the Xiaoqiang pose and path trajectory in rviz on the local window (the rviz cannot be directly opened in ssh), a distributed network configuration of ROS is needed, and the robot model package `xiaoqiang_udrf` is also required to be installed on the local machine. In summary: The local machine opens its own rviz, receives and displays the topic on the Xiaoqiang host, and the Xiaoqiang model data is obtained directly from the local. The specific process is as follows:

a. Open a terminal in the local, add Xiaoqiang's ip in the local hosts file

```
sudo gedit /etc/hosts
# Add to
xxx.xxx.xxx.xxx xiaoqiang-desktop # Please change xx to actual ip
```

b. Open a new terminal input

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
# Continue to execute
rostopic list
```

If you can see Xiaoqiang's topic, it means the configuration is successful.

d. Users who installed Xiaoqiang system image do not need perform this step. Install the model package, update the local ROS package environment variable, because the model data needs to be read from the local

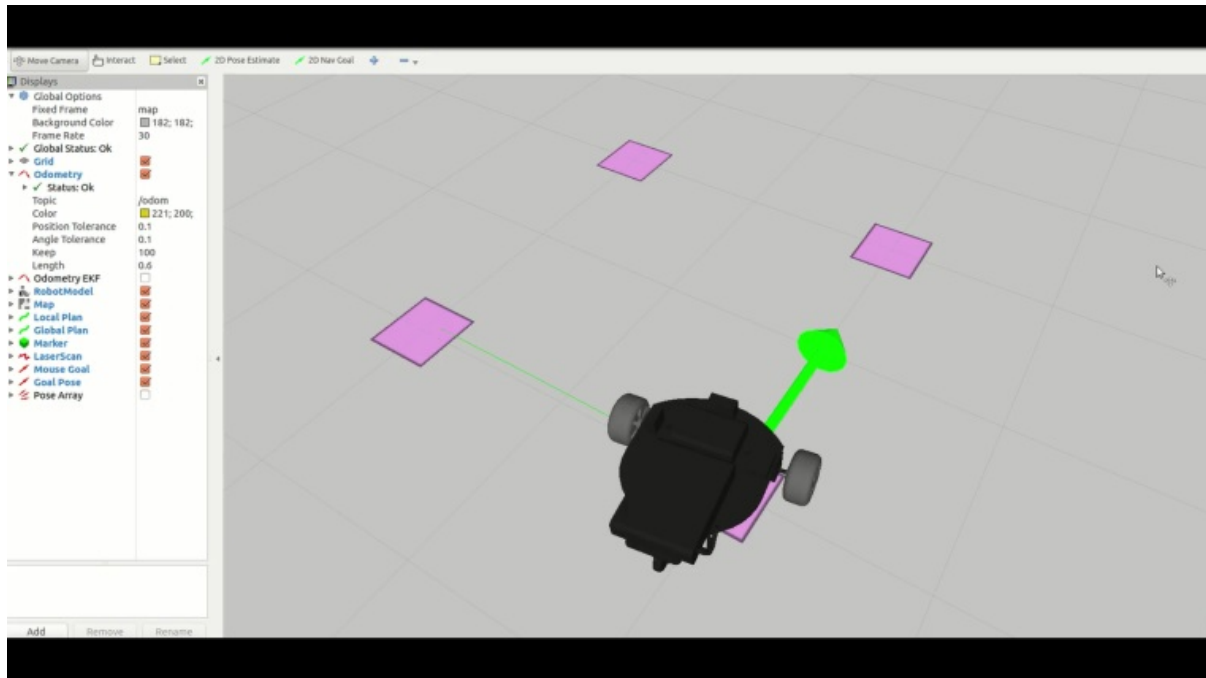
```
mkdir ~/Documents/ros/src
cd ~/Documents/ros/src
catkin_init_workspace
git clone https://github.com/BlueWhaleRobot/xiaoqiang_udrf.git
#Xiaoqiang pro user switches to the master branch
cd xiaoqiang_udrf
git checkout master
#Xiaoqiang mini user switches to mini branch
cd xiaoqiang_udrf
git checkout mini
# xiaoqiang XQ5 user switches to lungu branch
cd xiaoqiang_udrf
git checkout lungu
# Completion of the installation
cd ..
cd ..
catkin_make
```

e. Open the rviz.

```
export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
source ~/Documents/ros/devel/setup.sh
rviz
```

When the window opens, click on file->open in the upper left corner and select the

`/home/xiaoqiang/Documents/ros/src/nav_test/config/nav.rviz` file in Xiaoqiang. At this time, the interface should be displayed as shown below. Please refer to the previous tutorial for how to access the files on the Xiaoqiang host.



3. Complete the last operation in the remote host ssh window

```
roslaunch nav_test square.py
```

If Xiaoqiang cannot move, check if the platform driver is working properly. You can enter `rostopic echo /system_monitor/report` If there is a voltage display, the platform drive is normal. 如If it still cannot be moved, you can check whether Xiaoqiang infrared sensor is triggered or not. The infrared sensor will glow red when triggered.

4. Now you can see the video effect at the beginning of the article in rviz, enjoy it!

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(5\) xiaoqiang remote control app for Android](#)
 - [xiaoqiang remote control app Android version](#)
 - [Troubleshooting](#)

xiaoqiang tutorial (5) xiaoqiang remote control app for Android

[Xiaoqiang Homepage](#)

xiaoqiang remote control app Android version

Client with remote control and small screen image display function [xiaoqiang-with-control.apk](#)



Client without the remote control function, with full screen display [xiaoqiang-no-control.apk](#)



Usage:

1. Ensure that Xiaoqiang and the remote control handset are in the same LAN
2. Start the server program on Xiaoqiang
3. Open the app. If everything is normal, you can see Xiaoqiang's voltage display and Xiaoqiang image data. If there is no data you can try clicking the reconnect button.

Troubleshooting

Q: Unable to connect to xiaoqiang after app launched

A: It may be because Xiaoqiang and your phone are not in the same LAN. It is also possible that Xiaoqiang's server-side program does not start. You can enter `sudo service startup restart` to restart the service program and try again.

Q: Unable to remote control after successful connection

A: Check whether the driver is operating normally. Check whether the serial port USB of xiaoqiang is properly connected. Then enter `rostopic echo /system_monitor/report`. If the voltage is normal, then the platform is normal. If the voltage does not display properly, restart the service and see `sudo service startup restart`. If the voltage is normal but still can't move, check if infrared sensors triggered. The infrared sensor will glow red when triggered.

Q: No video transmission after successful connection

A: Check if the camera USB is connected properly. Then restart the service and try again `sudo service startup restart`

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(6\) xiaoqiang remote control Windows client](#)
 - [xiaoqiang remote control Windows client](#)
 - [Troubleshooting](#)

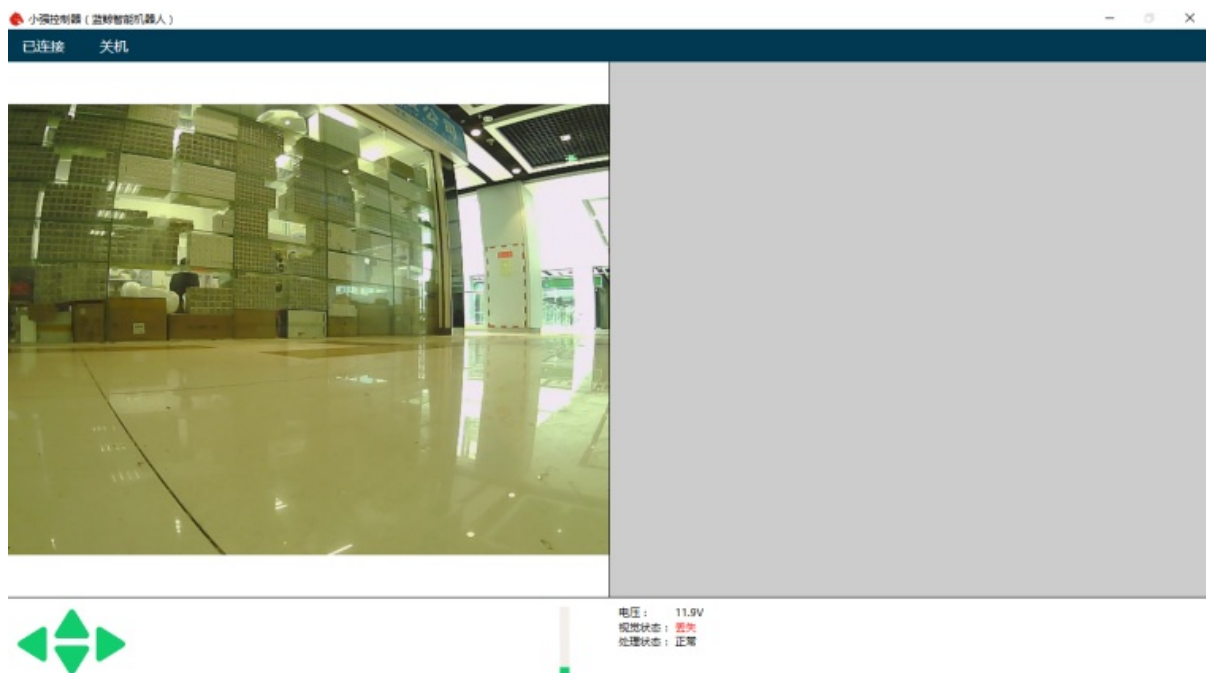
xiaoqiang tutorial (6) xiaoqiang remote control Windows client

[Xiaoqiang Homepage](#)

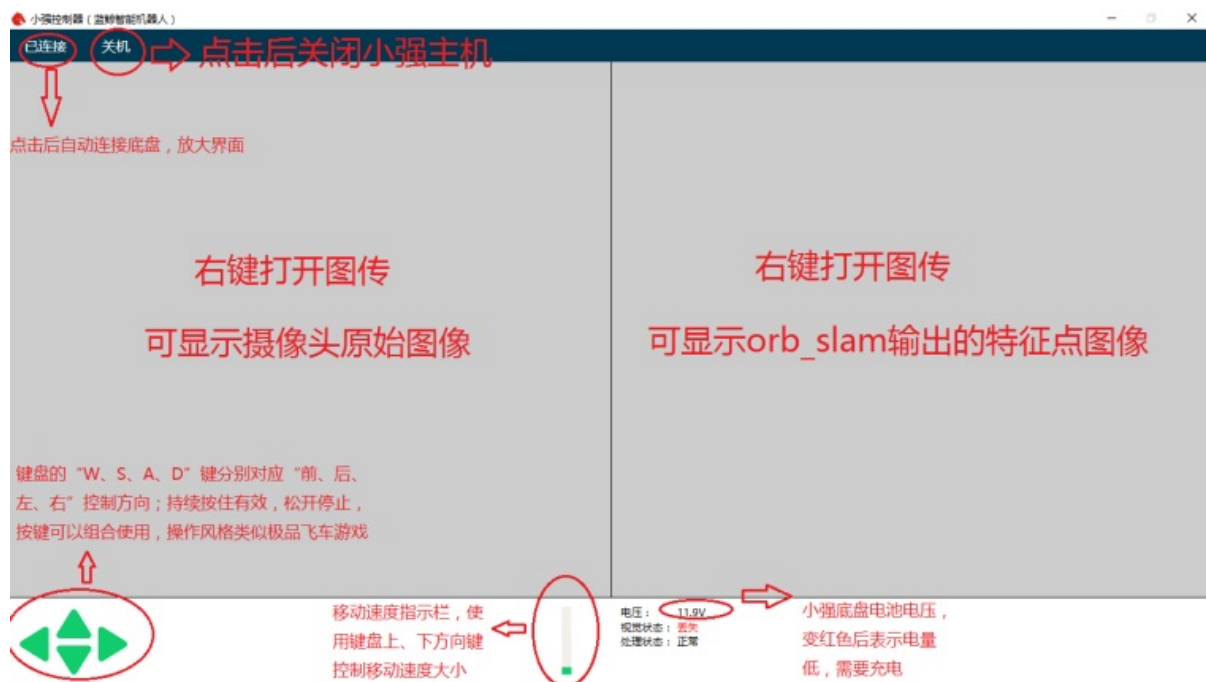
xiaoqiang remote control Windows client

The windows client has merged with Galileo navigation system client, please refer to [Galileo navigation system user manual](#)

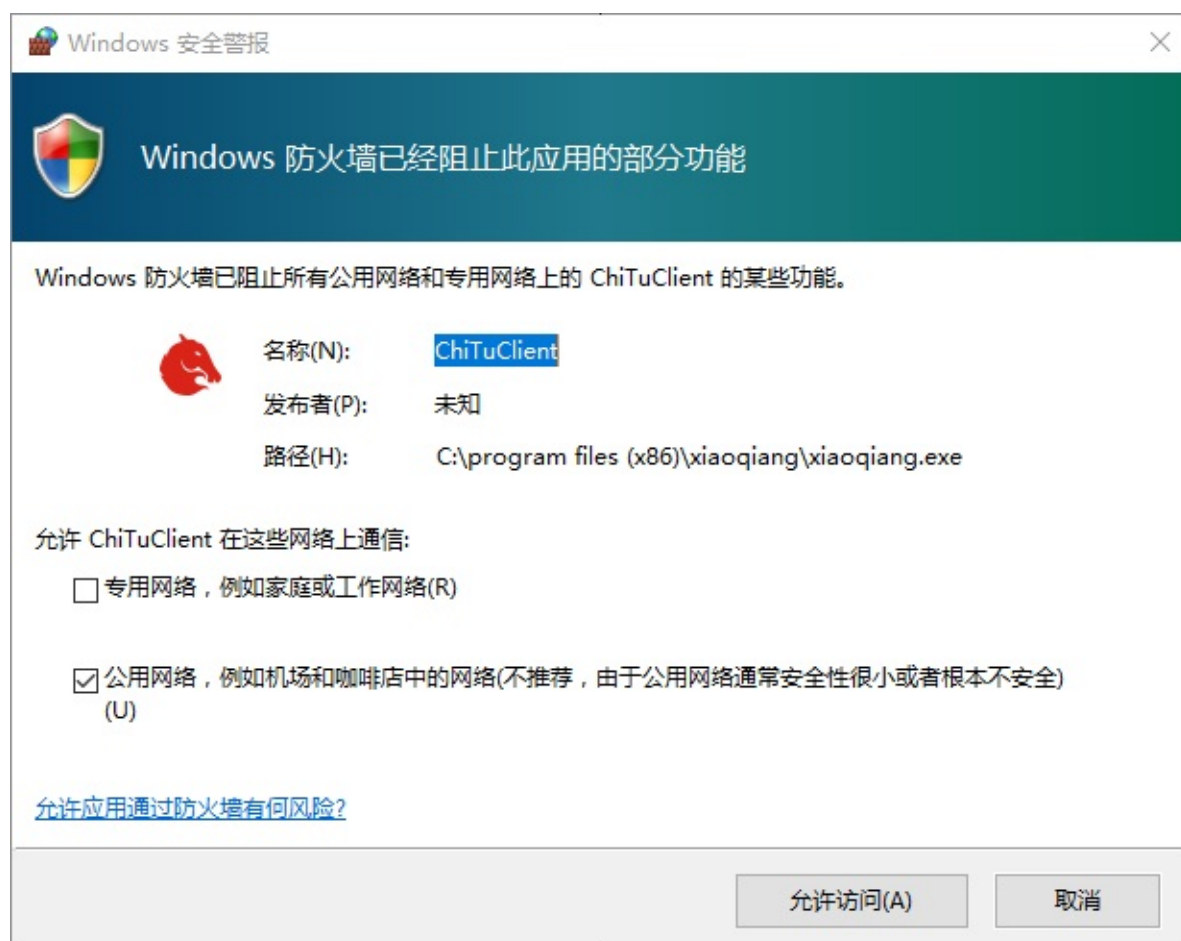
The software interface is shown below



[Software installation package download](#) Download and double click to install. Follow the instructions and click next all the way.



The first time you start the client, you will see the following prompts. Please tick "Private network and public network" and click to select "Allow access."



Troubleshooting

Q: Unable to connect after app launched

A: It may be because Xiaoqiang and your computer are not in the same LAN. It is also possible that Xiaoqiang's server-side program does not start. You can enter `sudo service startup restart` to restart the server and try again.

Q: Unable to remote control after successful connection

A: Check whether driver is working properly. Check whether the serial port USB of the xiaoqiang is properly connected. Then enter `rostopic echo /system_monitor/report` , If the voltage shows normal, then the platform is normal. If not you can try to restart your service. `sudo service startup restart` . If the voltage is normal but still can't move, check if infrared sensors triggered. The infrared sensor will glow red when triggered.

Q: No video transmission after successful connection

A: Check if the camera USB is connected properly. Then restart the service and try again.”`sudo service startup restart`”. enter `rostopic echo /system_monitor/report` , If imageStatus is True, shows that the image data is normal. If there is still no image displayed at this time, the client installation is abnormal. Please check the installation steps of the client.

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(7\) use PS3 joystick to control xiaoqiang](#)
 - [use PS3 joystick to control Xiaoqiang](#)
 - [Steps:](#)
 - [Pattern play:](#)

xiaoqiang tutorial (7) use PS3 joystick to control xiaoqiang

[Xiaoqiang Homepage](#)

use PS3 joystick to control Xiaoqiang

Principle: This tutorial involves 3 packages, [ps3joy](#) is responsible for converting ps3 bluetooth accept signals to standard linux devices (/dev/input/js0). The [joy_node](#) node is responsible for converting the above joy device data into the joy data type in ros. [turtlebot_teleop_joy](#) is responsible for converting the above joy data topic into motion instructions `/cmd_vel`

Steps:

0. The first time you use the joystick, you need to bind the joystick to the Bluetooth receiver. You can start directly from step 1 if you have bind the device before.

If you bought the joystick and xiaoqiang at the same time, Then we have already bind the device for you.

The binding method refers to `Quick use method step 1` in this [article](#)

1. Start ps3joy and bind the ps3 joystick to the bluetooth receiver

```
# Make sure the Bluetooth receiver is plugged into the host usb port
sudo bash
roslaunch ps3joy ps3joyfake_node.py
```

The following prompts appear normally

```
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
.
```

If the following error is indicated Press the handle pairing key in the following figure



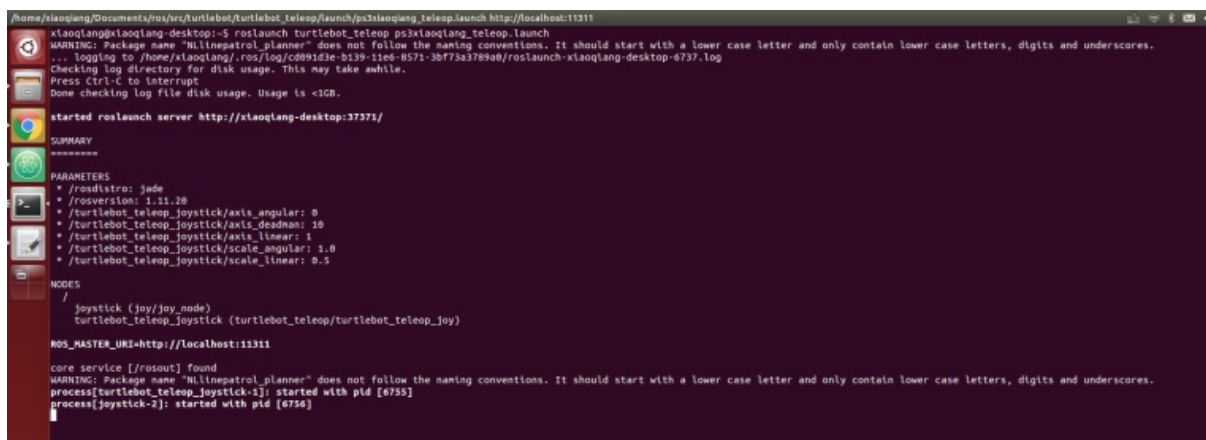
If the pairing is successful, the upper window will output a result similar to the following

```
root@xiaoqiang-desktop:~# rosrund 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. Start joy_node and turtlebot_teleop_joy

```
roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
```

After the normal start as shown in the figure below



```
home/xiaoqiang/Documents/ros/src/turtlebot/turtlebot_teleop/launch/ps3fakexiaoqiang_teleop.launch http://localhost:11311
xiaoqiang@xiaoqiang-desktop:~$ roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
WARNING: Package name "Nllinepatrol_planner" 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/c0891d3e-b139-11e6-8571-30f73a3789a0/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 <1GB.

started roslaunch server http://xiaoqiang-desktop:37371/

SUMMARY
=====
PARAMETERS
 * /roscdastro: jade
 * /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: 1.0
 * /turtlebot_teleop_joystick/scale_linear: 0.5

NODES
 /
  joystick (joy/joy_node)
  turtlebot_teleop_joystick (turtlebot_teleop/turtlebot_teleop_joy)

ROS_MASTER_URI=http://localhost:11311

core service [/rosout] found
WARNING: Package name "Nllinepatrol_planner" 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]
```

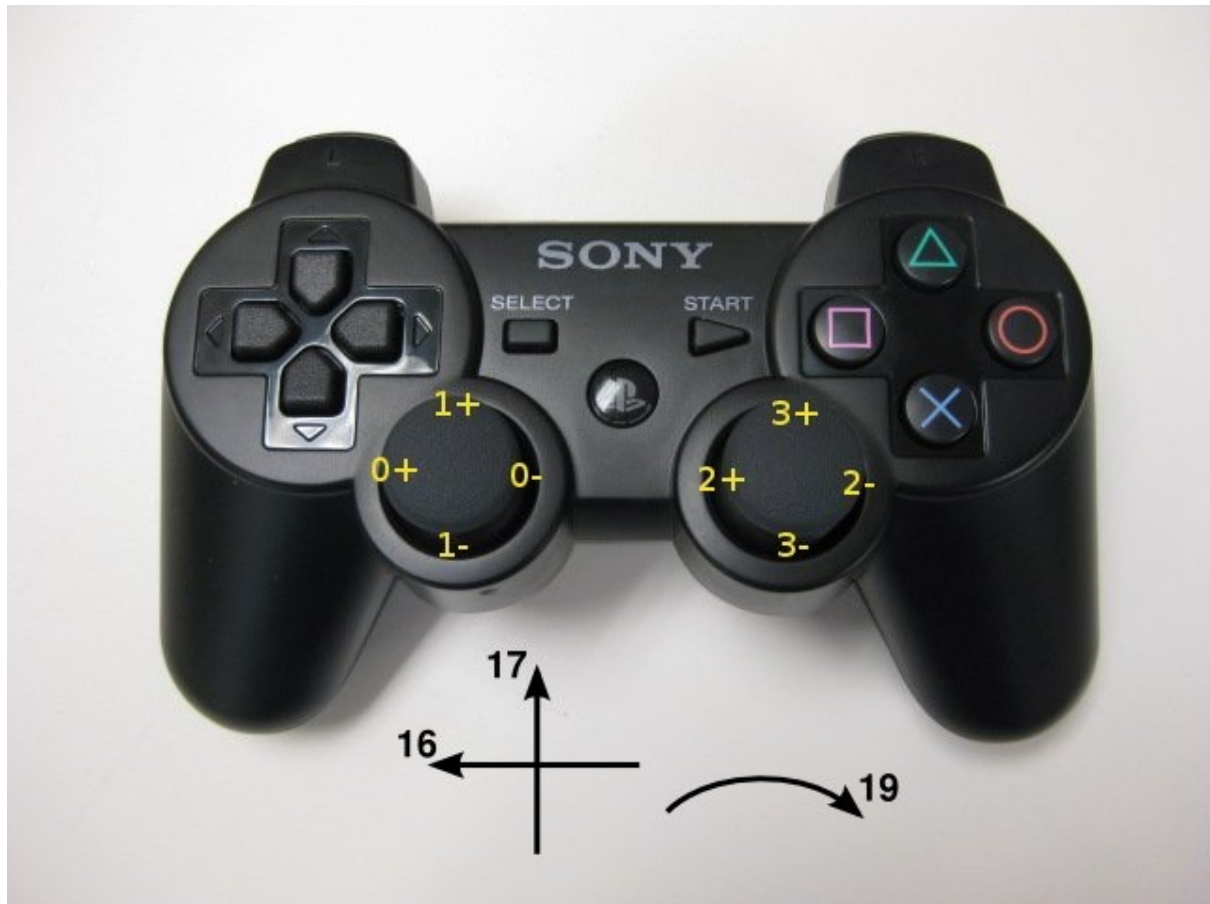
ps3fakexiaoqiang_teleop.launch file content is as follows

```
<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>
```

The parameters in the above launch file correspond to: Linear speed maximum (scale_linear), Angular speed maximum (scale_angular), Throttle key (axis_deadman), Forward and backward axis (axis_linear), Left and right shafts (axis_angular), Platform infrared enable key (axis_enbar), Platform infrared close button (axis_disenbar).

3. Hold down the handle throttle key (key 10 in the figure below) and now use the left pusher to control the platform's back and forth movement and steering (-1、 +1 rocker in the image below)





According to these button numbers, the relevant parameters in the launch file can be modified to change the button mapping relationship

Pattern play:

Purchase ps3 mobile phone holder, Android mobile phone loaded [Xiaoqiang picture app](#), so you can achieve remote control.



[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(8\) kinect v1 ROS driver test and installation](#)
 - [kinect v1 ROS driver test and installation](#)
 - [1. libfreenect test](#)
 - [2. ROS drive test](#)
 - [3. The following describes the ros driver installation procedure of kinect v1](#)
 - [Need to install three software:](#)

xiaoqiang tutorial (8) kinect v1 ROS driver test and installation

[Xiaoqiang Homepage](#)

kinect v1 ROS driver test and installation

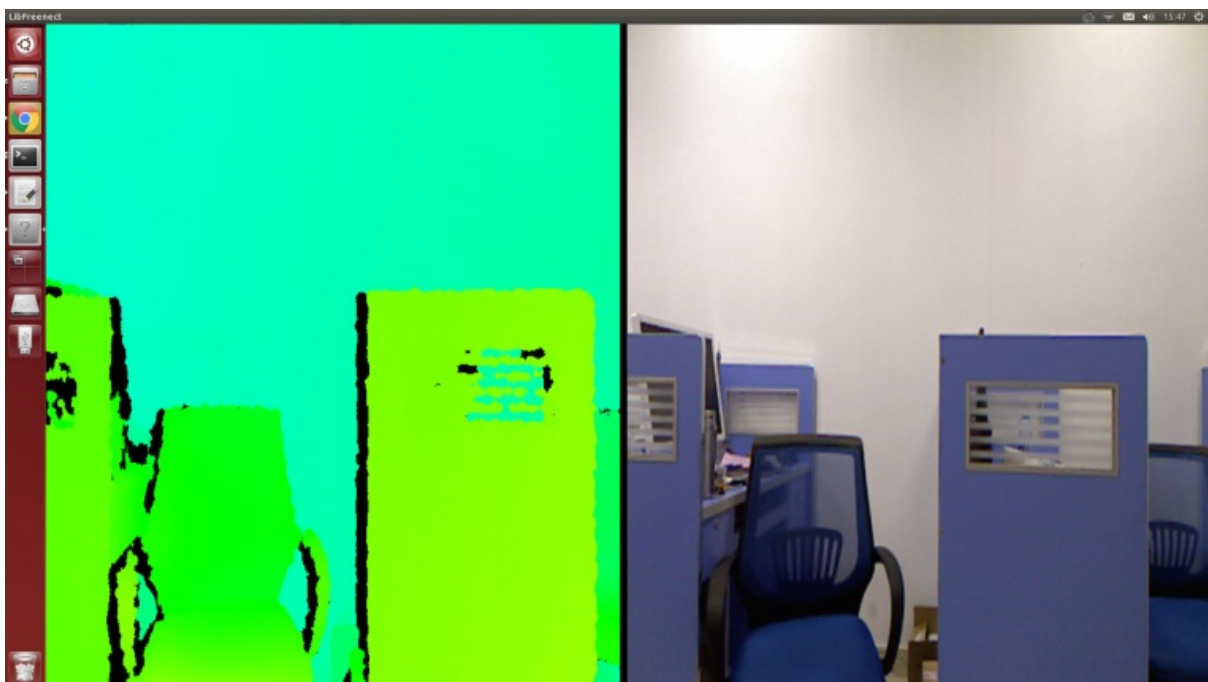
Xiaoqiang outputs a 12V power supply (DC head, labeled “kinect power supply”) for kinect power supply.

1. libfreenect test

Connect the Xiaoqiang host with monitor and keyboard and open a new terminal input on the Xiaoqiang host

```
freenect-glview
```

You can see the similar figure below



2. ROS drive test

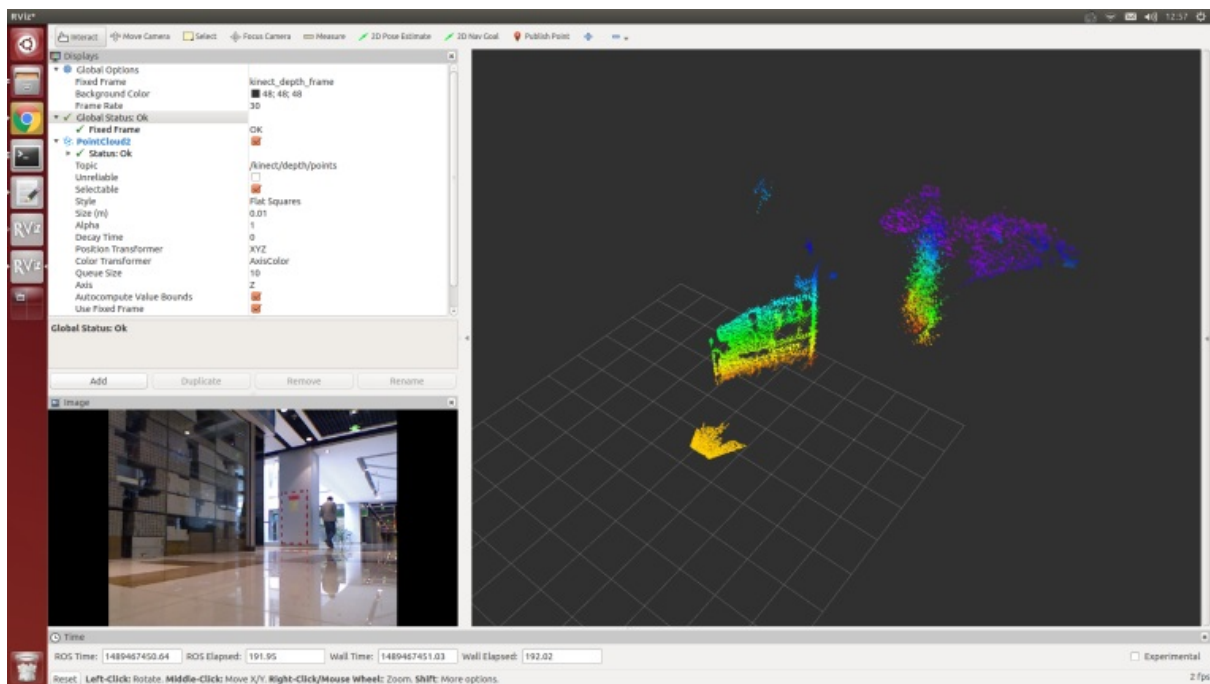
Close the program in step 1, open a new terminal, use `freenect_launch` to start the relevant kinect node

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

Open a new window to open `rviz`

```
rviz
```

Select the content to be displayed, such as kinect rgb image and depth point cloud, the display result is as follows



The settings of kinect functions is in

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

```
<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>
```

Turn on or off the feature by setting true or false

3. The following describes the ros driver installation procedure of kienct v1

If you use xiaoqiang system image, then you don't need to install the following drivers. These drivers were already install.

Need to install three software:

a.libfreenect

b.rgbd_launch

c.freenect_stack

a. libfreenect

First connect the kinect v1 to the xiaoqiang host, then open a terminal and enter the following code

```
cd Documents

sudo apt-get install git-core cmake freeglut3-dev pkg-config build-essential libxmu-dev libxi-dev libusb-1.0-0-dev

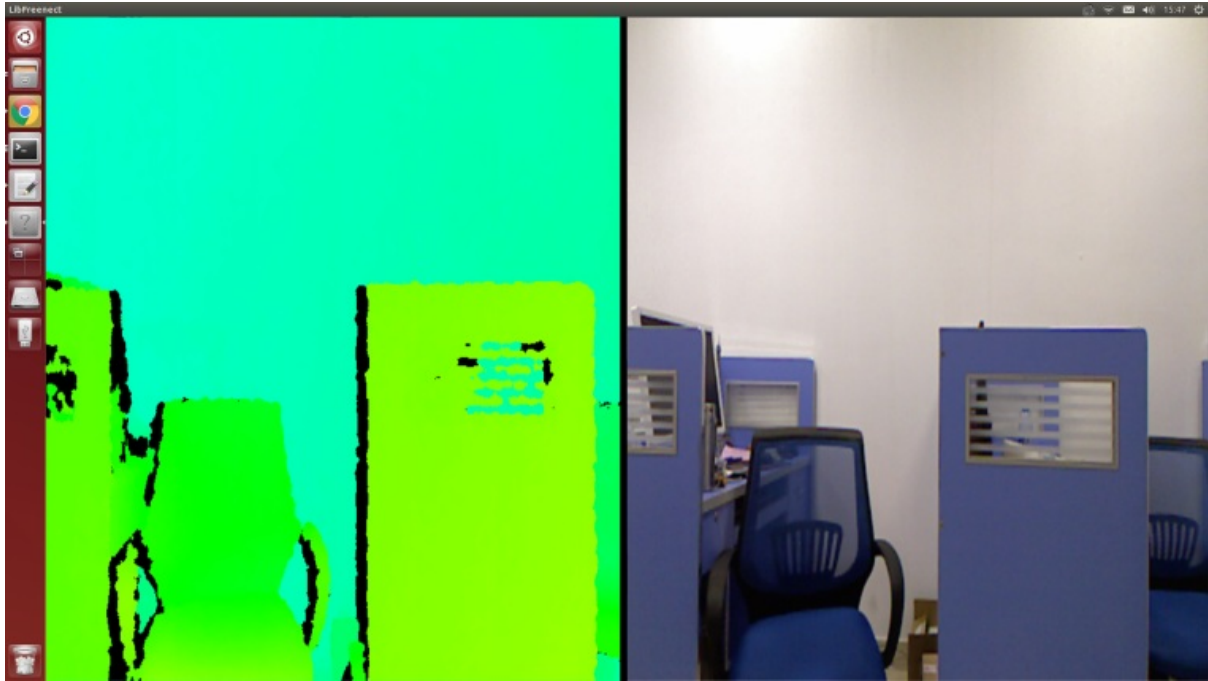
git clone git://github.com/OpenKinect/libfreenect.git
cd libfreenect
mkdir build
cd build

//The key point is, the following configuration will enable kinect audio and solve the installation path problem

cmake .. -DCMAKE_INSTALL_RPATH:STRING="/usr/local/bin;/usr/local/lib" -DBUILD_REDIST_PACKAGE=OFF

make
sudo make install
sudo ldconfig /usr/local/lib64/
sudo freenect-glview
```

You should see the output image of the kinect now , and then configure the peripheral privilege configuration



```
sudo adduser $xiaoqiang video
#Please replace xiaoqiang with your own computer's account name
```

Add a udev rule, first open the 51-kinect.rules file

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

Copy the contents below and save and exit

```
# 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"
```

After logging off the user and reentering the system, you can now enable kinect directly without sudo.

```
freenect-glvie
```

b. Install rgbd_launch

`rgbd_launch` contains generic launch files required by the driver installation package `openni_launch` or `freenect_launch`. There are two main launch files:

1. `processing.launch.xml` : Install a series of nodelets to process data from the RGB-D driver `openni_camera` or `freenect_camera`, You can also set parameters to simplify processing nodelets images.
2. `kinect_frames.launch` : Install the tf tree for kinect. The file can also be started internally from `openni_launch` or `freenect_launch`.

The `rgbd_launch` file contains multiple distributed launch files. But only `processing.launch.xml` can be modified externally.

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

c. Install `freenect_stack`

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

d. Driver installation is complete, you can now use kinect in ROS, for example, in `rviz` view kinect output point cloud, refer to step 2 at the beginning of this section.

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(9\) use rostopic to control kinect tilt angle](#)
 - [use rostopic control kinect tilt angle](#)
 - Prepared work:
 - Steps:

xiaoqiang tutorial (9) use rostopic to control kinect tilt angle

[Xiaoqiang Homepage](#)

use rostopic control kinect tilt angle

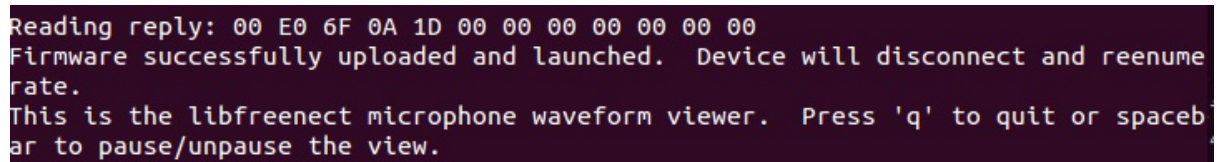
Prepared work:

Please check the kinect version, indicated on the kinect base label. For the users of the model1473, due to the defect of the drive (which does not affect other functions of the kinect, only the motor), the following operations need to be performed first, and the model1414 user can directly skip.

ssh login xiaoqiang host

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

If the following figure appears, close the above command and continue the tutorial



```
Reading reply: 00 E0 6F 0A 1D 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 spaceb
ar to pause/unpause the view.
```

Steps:

1. Open a new window in the local virtual machine, start the freenect_stack driver

ssh login Xiaoqiang host

```
ssh xiaoqiang@192.168.0.xxx -X
roslaunch freenect_launch freenect-xyz.launch
```

The normal startup will appear below, if there is a red error (driver defect), please update the `freenect_stack` program through git pull.

```
[ INFO] [1477406854.865362860]: Number devices connected: 1
[ INFO] [1477406854.865453964]: 1. device on bus 000:00 is a Xbox NUI Camera (2e) 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/camera_info/rgb_A70774703536333A.yaml not found.
[ WARN] [1477406856.975681775]: Using default parameters for RGB camera calibration.
[ WARN] [1477406856.975777743]: Camera calibration file /home/xiaoqiang/.ros/camera_info/depth_A70774703536333A.yaml not found.
[ WARN] [1477406856.975820217]: Using default parameters for IR camera calibration.
[ INFO] [1477406859.826134535]: Stopping device RGB and Depth stream flush.
```

2. Open a new window in the local virtual machine, release the motor angle control command

ssh login Xiaoqiang host

```
ssh xiaoqiang@192.168.0.xxx -X
rostopic pub /set_tilt_degree std_msgs/Int16 '{data: -20}' -1
```

If all goes well, you can now see that the elevation of the kinect keeps getting smaller. The `{data: -20}` number in the above command represents the angle and can be set to an integer between 30 and -30. Insert the kinect cable into the host's blue Color usb3.0 port so that kinect can work properly.

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(10\) use kinect for autonomous mobile and obstacle avoidance](#)
 - [principle:](#)
 - [Steps :](#)
 - [1. Configure the hosts file of the host and the local virtual machine so that the two computer can access the corresponding ros data.](#)
 - [2. Open three windows in the local virtual machine, respectively ssh login Xiaoqiang host, start the relevant software package mentioned in the principle section](#)
 - [3. Open a new window in the local virtual machine and start rviz](#)
 - [FAQ](#)

xiaoqiang tutorial (10) use kinect for autonomous mobile and obstacle avoidance

[Xiaoqiang Homepage](#)

Click to watch the demo video



principle:

The `freenct_stack` package provides a kinect driver, and its published point cloud is transformed into an obstacle grid distribution image by `image_pipeline`. After the `nav_test` software package starts the navigation program, it analyzes the obstacle distribution map automatically, and then moves autonomously according to the target navigation point published by the rviz.

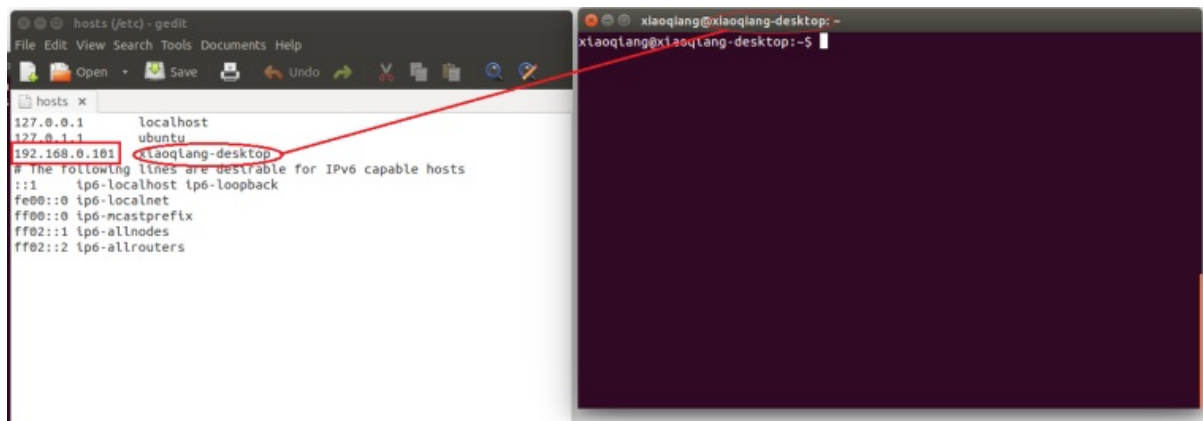
Steps:

1. Configure the hosts file of the host and the local virtual machine so that the two computer can access the corresponding ros data.

1.a Configure the local virtual machine

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


Replace the ip address in the figure below with the actual value of the platform

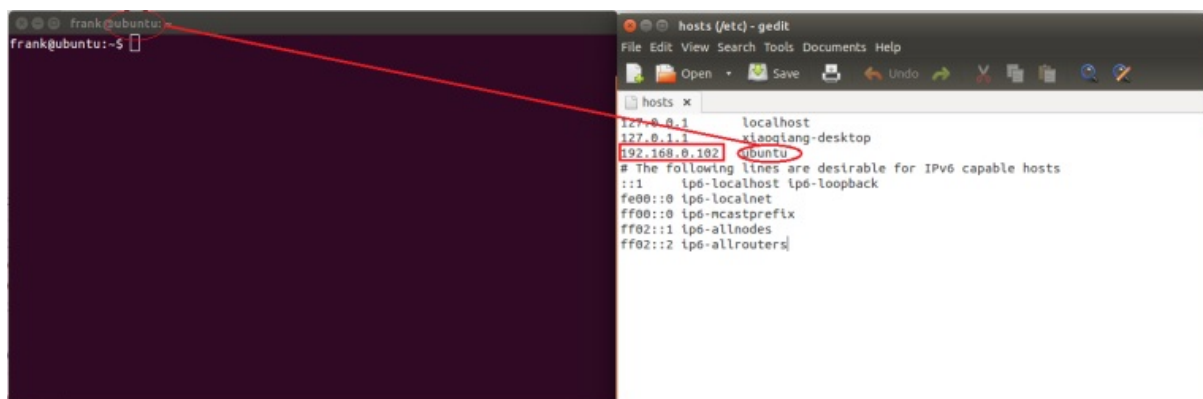


1.b Configuration Xiaoqiang Host

LAN ssh login Xiaoqiang host

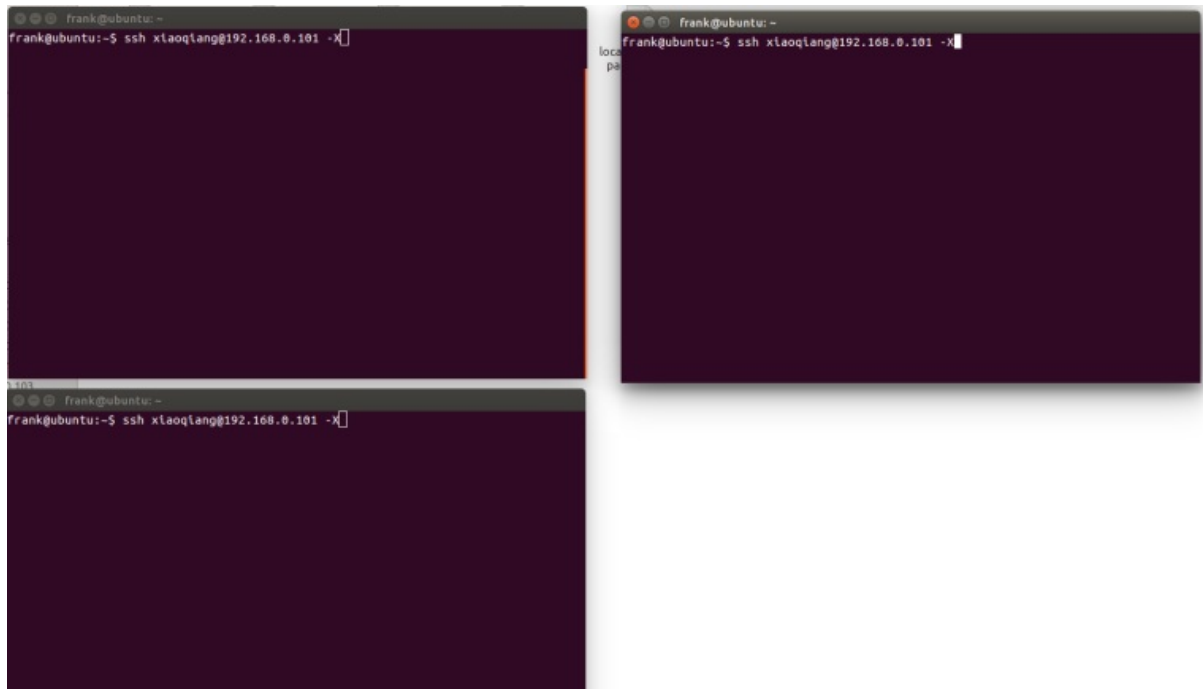
```
ssh xiaoqiang@192.168.0.101 -X
sudo gedit /etc/hosts
```

Replace the ip address in the following figure with the actual value of the virtual machine and change the host name to the virtual machine name



2. Open three windows in the local virtual machine, respectively ssh login Xiaoqiang host, start the relevant software package mentioned in the principle section

2.a Ssh login



2.b Start the kinect driver in the first window

For kinect v1

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

For Kinect v2

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

For astrapro

```
roslaunch astra_launch astrapro.launch
```

2.c Set kinect angle in the second window, this angle is not arbitrary

For kinect v1

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

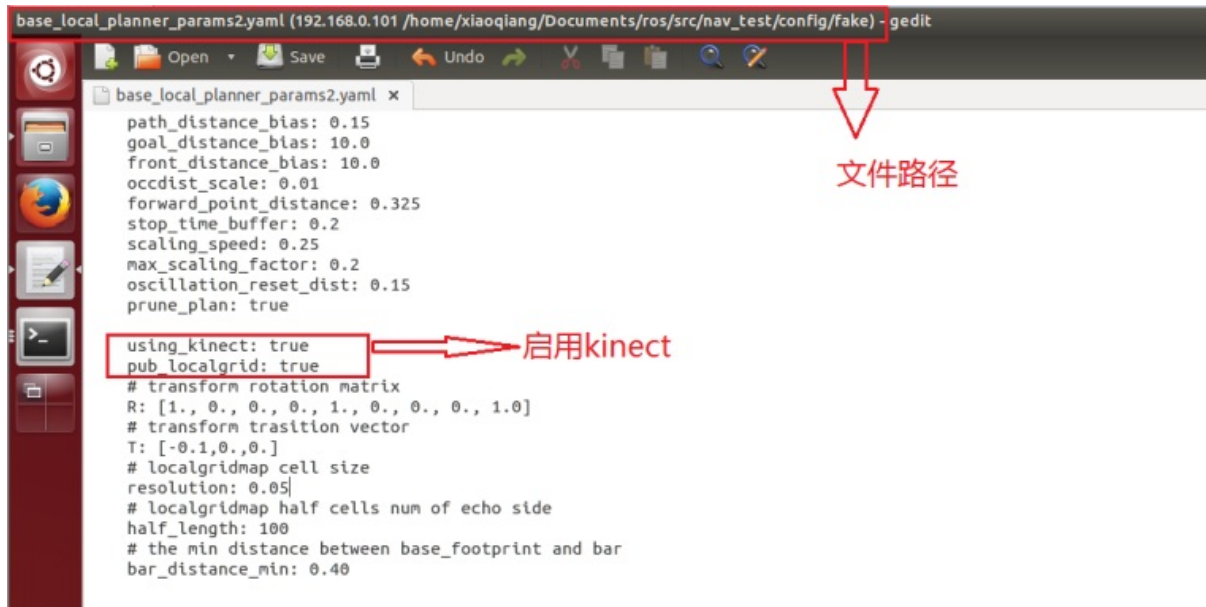
For Kinect v2

Since the kinect v2 does not have a pitch angle motor, manually adjust the kinect v2 to the maximum depression angle

For astrapro

Since the astrapro does not have a pitch angle motor, manually adjust the astrapro to the maximum depression angle.

2.d Edit the platform navigator configuration file /home/xiaoqiang/Documents/ros/src/nav_test/config/fake/base_local_planner_params2.yaml to enable kinect



```
base_local_planner_params2.yaml (192.168.0.101 /home/xiaoqiang/Documents/ros/src/nav_test/config/fake) - gedit
base_local_planner_params2.yaml x
path_distance_bias: 0.15
goal_distance_bias: 10.0
front_distance_bias: 10.0
occdist_scale: 0.01
forward_point_distance: 0.325
stop_time_buffer: 0.2
scaling_speed: 0.25
max_scaling_factor: 0.2
oscillation_reset_dist: 0.15
prune_plan: true

using_kinect: true
pub localgrid: true
# transform rotation matrix
R: [1., 0., 0., 0., 1., 0., 0., 0., 1.0]
# transform translation vector
T: [-0.1,0.,0.]
# localgridmap cell size
resolution: 0.05]
# localgridmap half cells num of echo side
half_length: 100
# the min distance between base_footprint and bar
bar_distance_min: 0.40
```

2.e Launch the platform navigation program in the third window

```
roslaunch nav_test fake_move_base_blank_map.launch
```

2.f If all normal, there will be an interface similar to the following figure. All configurations on xiaoqiang have completed.

```

/home/xiaoqiang/Documents/ros/src/freenect_stack/freenect_launch/launch/kinect-xyz
[ INFO] [1477451417.068842198]: Number devices connected: 1
[ INFO] [1477451417.069079865]: 1. device on bus 000:00 is a Xbox NUI Camera (28e) from Microsoft (45e) with serial id 'A70774703536333A'
[ INFO] [1477451417.070199975]: Searching for device with index = 1
[ INFO] [1477451417.63566096]: flushDevice
[ INFO] [1477451417.636953796]: Starting a 3s RGB and Depth stream flush.
[ INFO] [1477451417.636452059]: Opened 'Xbox NUI Camera' on bus 0:0 with serial number 'A70774703536333A'
[ WARN] [1477451418.297373672]: Could not find any compatible depth output mode for 1. Falling back to default depth output mode 1.
[ INFO] [1477451420.771565812]: Stopping device RGB and Depth stream flush.
[ INFO] [1477451420.780670853]: rgb_frame_id = 'kinect_rgb_optical_frame'
[ INFO] [1477451420.780713115]: depth_frame_id = 'kinect_depth_optical_frame'
[ WARN] [1477451420.792224733]: Camera calibration file /home/xiaoqiang/.ros/camera_info/rgb_A70774703536333A.yaml not found.
[ WARN] [1477451420.792270926]: Using default parameters for RGB camera calibration.
[ WARN] [1477451420.792307384]: Camera calibration file /home/xiaoqiang/.ros/camera_info/depth_A70774703536333A.yaml not found.
[ WARN] [1477451420.792332082]: Using default parameters for IR camera calibration.

/home/xiaoqiang/Documents/ros/src/nav_test/launch/fake_move_base_blank_map.launch
[ INFO] [1477451503.621722491]: Requesting the map...
[ INFO] [1477451503.826107431]: Resizing costmap to 600 X 600 at 0.100000 m/pix
[ INFO] [1477451503.925452905]: Received a 600 X 600 map at 0.100000 m/pix
[ INFO] [1477451503.931700165]: Using plugin "obstacle_layer"
[ INFO] [1477451503.934483974]: Subscribed to Topics: scan
[ INFO] [1477451503.959462249]: Using plugin "inflation_layer"
[ INFO] [1477451504.037680709]: Loading from pre-hydro parameter style
[ INFO] [1477451504.056195124]: Using plugin "static_layer"
[ INFO] [1477451504.081866143]: Requesting the map...
[ INFO] [1477451504.084554880]: Resizing costmap to 600 X 600 at 0.100000 m/pix
[ INFO] [1477451504.184278987]: Received a 600 X 600 map at 0.100000 m/pix
[ INFO] [1477451504.188645977]: Using plugin "obstacle_layer"
[ INFO] [1477451504.190425875]: Subscribed to Topics: scan
[ INFO] [1477451504.213882631]: Using plugin "inflation_layer"
[ INFO] [1477451504.290588698]: ODOM SET!
[ INFO] [1477451504.296691310]: Created local_planner addwa_local_planner/ADDWAPlannerROS
[ INFO] [1477451504.845235067]: Sin period is set to 0.07
[ INFO] [1477451505.283206996]: Recovery behavior will clear layer obstacles
[ INFO] [1477451505.309189793]: Recovery behavior will clear layer obstacles
[ INFO] [1477451505.341844221]: odom received! 1.539172,-0.666669

```

3. Open a new window in the local virtual machine and start rviz

3.a After joining ros LAN, open rviz

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

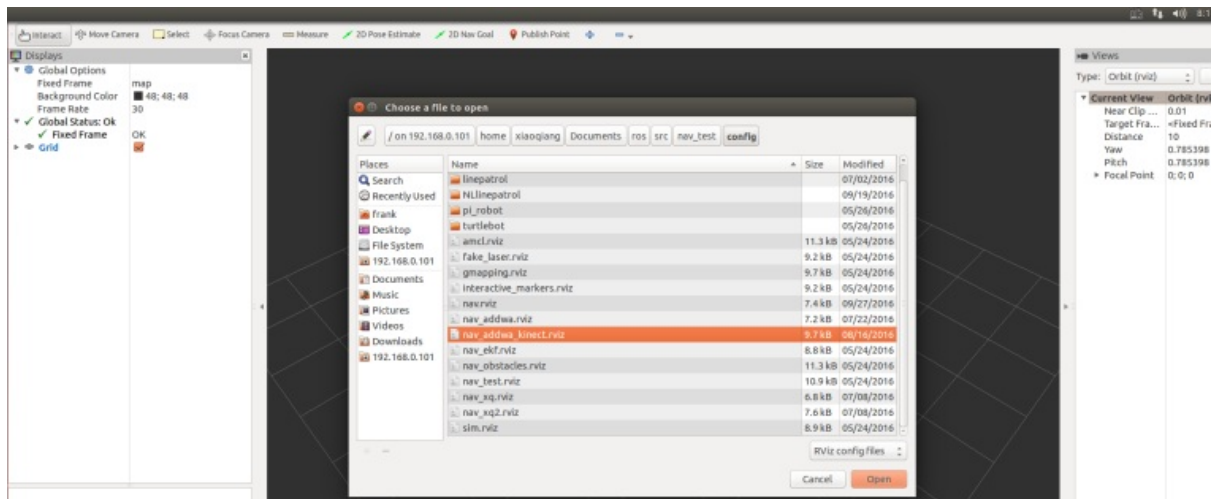
Note: If Xiaoqiang's model cannot be displayed, Xiaoqiang model package should be installed locally.

```

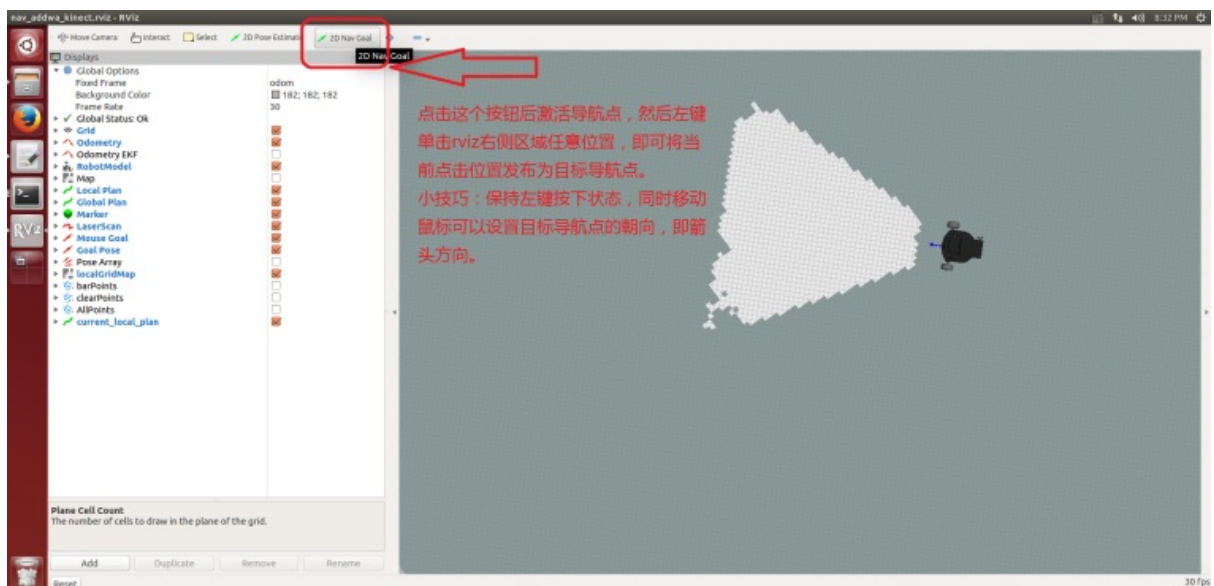
frank@ubuntu: ~
frank@ubuntu:~$ export ROS_MASTER_URI=http://xiaoqiang-desktop:11311
frank@ubuntu:~$ rviz

```

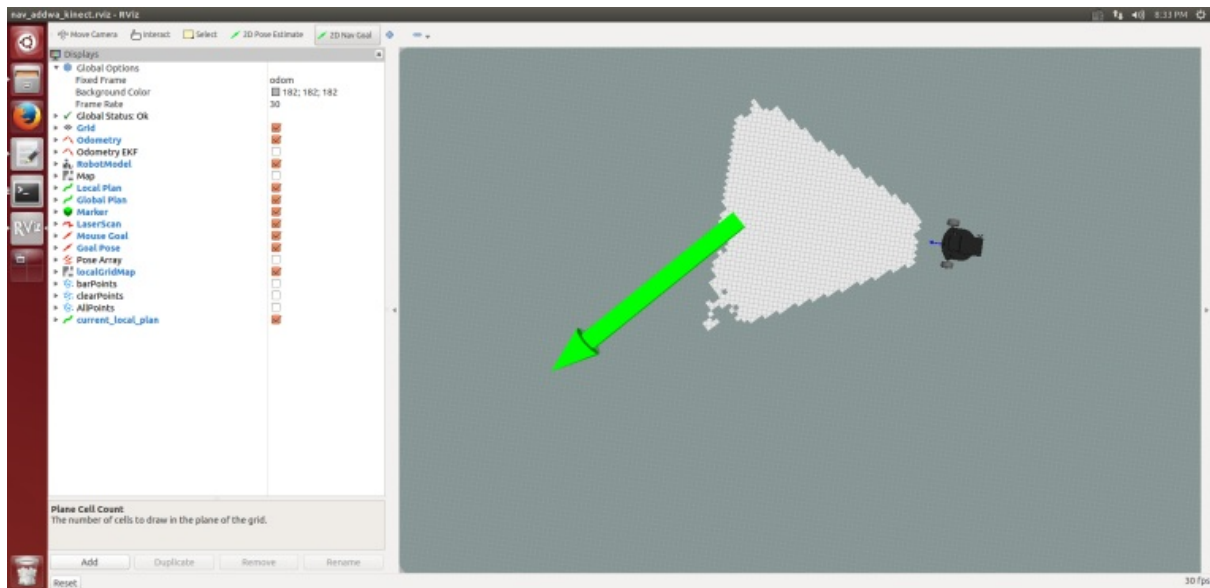
3.b Click open config in the upper left corner of the rviz interface and select the /home/xiaoqiang/Documents/ros/src/nav_test/config/nav_addwa_kinect.rviz configuration file on Xiaoqiang's host.



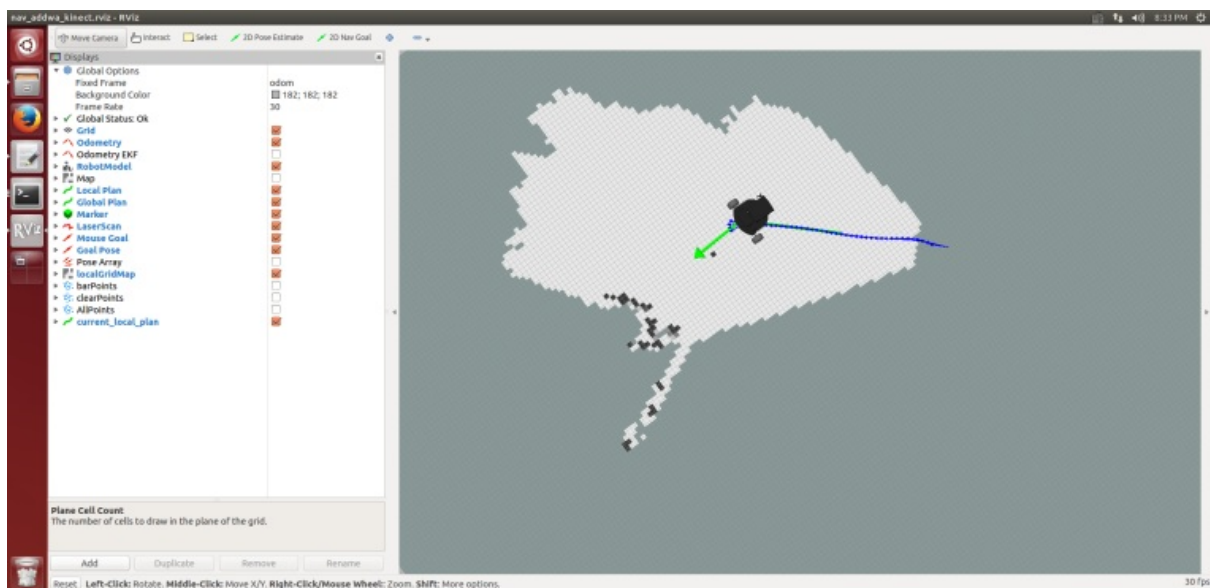
3.c Normally, a picture similar to the one below will appear in rviz. Now that all configurations have been completed, the next step is to publish navigation targets.



3.d Randomly publish a target point, Xiaoqiang will start autonomous movement



3.e Xiaoqiang reaches the target point, please continue to try other positions. This tutorial ended.



Please insert the kinect data cable into the blue host USB3.0 port of xiaoqiang to ensure the normal operation of the driver. Model 1473 kinect users please refer to the preparation method in tutorial (9).

If xiaoqiang has been moving backwards, it can be seen in rviz that there is a black area in front of xiaoqiang, and the black area has been following Xiaoqiang's movement. This situation indicates that the kinect installation or kinect angle is not in right. Please check the kinect mounting bracket, remove the kinect film and re-execute the instructions in the tutorial.

FAQ

Q: Map shows normally and the car can also be remote controlled, although published the target point via rviz to the car, the car does not move to the target point.

A: In general, this problem is a network setup problem. This tutorial requires the hosts record to be set up between the xiaoqiang and the local machine, and to be able to access each other directly. After setting up the IP record, you can ping each other on the xiaoqiang and local computer to check the network connection. For example type following cmd on local machine

```
ping xiaoqiang-desktop
```

Enter the following cmd on xiaoqiang

```
ping ubuntu # Suppose your computer name is ubuntu
```

If we cannot ping between the two successfully, then you need check your network settings.

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(11\) kinect follow package turtlebot_follower](#)
 - [kinect follow package turtlebot_follower](#)

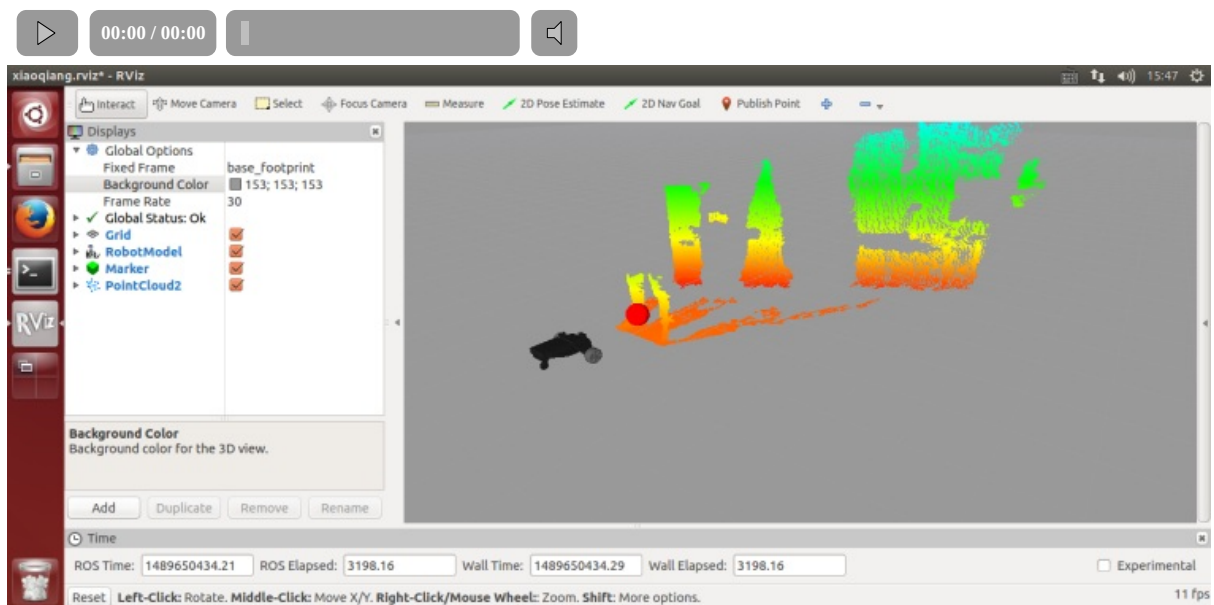
xiaoqiang tutorial (11) kinect follow package turtlebot_follower

[Xiaoqiang Homepage](#)

kinect follow package turtlebot_follower

`turtlebot_follower` uses the point cloud image feed back by the depth camera to calculate the coordinates of the point cloud in a certain area as the target following point, and controls the platform movement to achieve the following function according to this coordinate and the set safety distance.

After Xiaoqiang identifies the human legs, a red ball is set as the target point in the human leg position, demo video 1



1. Install the package

After ssh logs in to the host, enter the ros workspace, download and install the test package

```
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. Ensure that Xiaoqiang is 2 meters * 2 meters in front of the open space without debris, first check the kinect angle is horizontal, if not in the horizontal state, please turn off the platform power, start the `turtlebot_follower` package on the Xiaoqiang host

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

The following figure will appear after normal startup

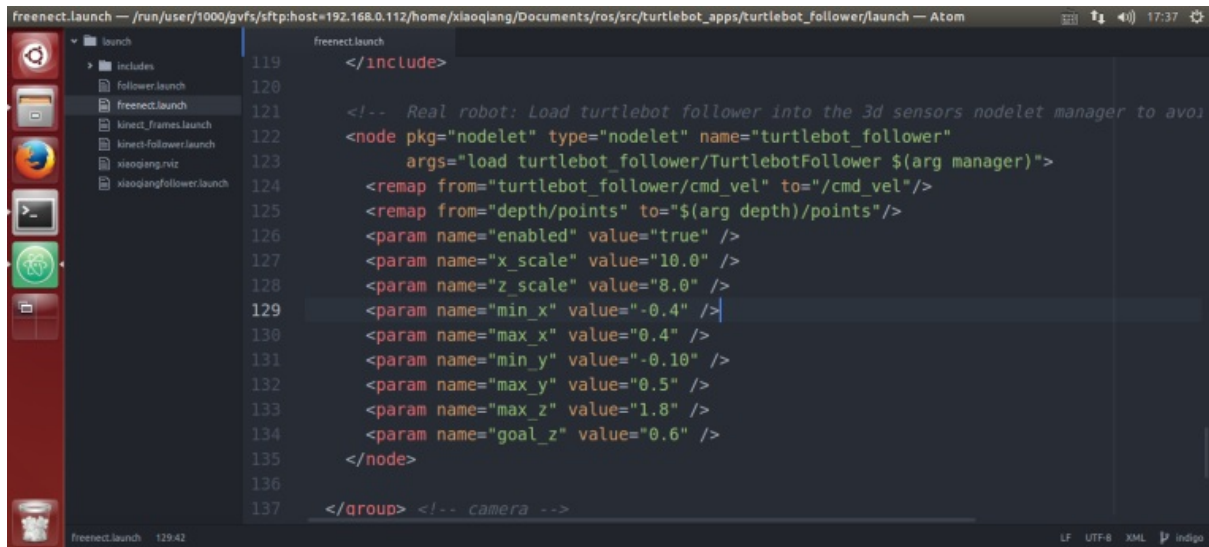
```
/home/xiaoqiang/Documents/ros/src/turtlebot_apps/turtlebot_follower/launch/xiaoqiangfollower.launch http://localhost:11311
process[kinect/depth_points-6]: started with pid [13698]
process[kinect/turtlebot_follower-7]: started with pid [13708]
process[kinect_base_link4-8]: started with pid [13719]
process[kinect_base_link-9]: started with pid [13736]
process[kinect_base_link1-10]: started with pid [13747]
process[kinect_base_link2-11]: started with pid [13761]
process[kinect_base_link3-12]: started with pid [13769]
process[switch-13]: started with pid [13780]
[INFO] [1489650293.791091418]: Initializing nodelet with 4 worker threads.
[INFO] [1489650295.098157185]: subdevs: 3
[INFO] [1489650295.099490753]: Number devices connected: 1
[INFO] [1489650295.099603613]: 1. device on bus 000:00 is a Xbox NUI Camera (2ae) from Microsoft (45e) with serial id 'B003647038201038'
[INFO] [1489650295.102317526]: Searching for device with Index = 1
[INFO] [1489650295.288207715]: flushDevice
[INFO] [1489650295.288456046]: Starting a 3s RGB and Depth stream flush.
[INFO] [1489650295.288907037]: Opened 'Xbox NUI Camera' on bus 0:0 with serial number 'B003647038201038'
[WARN] [1489650295.412495795]: Could not find any compatible depth output mode for 1. Falling back to default depth output mode 1.
[INFO] [1489650295.468590422]: rgb_frame_id = 'kinect_rgb_optical_frame'
[INFO] [1489650295.468890361]: depth_frame_id = 'kinect_depth_optical_frame'
[WARN] [1489650295.527469054]: Camera calibration file /home/xiaoqiang/.ros/camera_info/rgb_B003647038201038.yaml not found.
[WARN] [1489650295.527609885]: Using default parameters for RGB camera calibration.
[WARN] [1489650295.528251620]: Camera calibration file /home/xiaoqiang/.ros/camera_info/depth_B003647038201038.yaml not found.
[WARN] [1489650295.528331743]: Using default parameters for IR camera calibration.
[INFO] [1489650296.522221494]: Not enough points(0) detected, stopping the robot
[INFO] [1489650297.523311453]: Not enough points(0) detected, stopping the robot
[INFO] [1489650298.790135158]: Stopping device RGB and Depth stream flush.
[INFO] [1489650298.556709616]: Not enough points(0) detected, stopping the robot
[INFO] [1489650299.559501856]: Not enough points(0) detected, stopping the robot
[INFO] [1489650300.627012057]: Not enough points(0) detected, stopping the robot
```

1. At this point, someone enters Xiaoqiang's front view and activates Xiaoqiang's follow function. Xiaoqiang starts to move with the person's movement.

```
/home/xiaoqiang/Documents/ros/src/turtlebot_apps/turtlebot_follower/launch/xiaoqiangfollower.launch http://localhost:11311
[INFO] [1489650298.556709616]: Not enough points(0) detected, stopping the robot
[INFO] [1489650299.559501856]: Not enough points(0) detected, stopping the robot
[INFO] [1489650300.627012057]: Not enough points(0) detected, stopping the robot
[INFO] [1489650301.627176819]: Not enough points(0) detected, stopping the robot
[INFO] [1489650302.629076837]: Not enough points(0) detected, stopping the robot
[INFO] [1489650303.629516902]: Not enough points(0) detected, stopping the robot
[INFO] [1489650304.630824086]: Not enough points(0) detected, stopping the robot
[INFO] [1489650305.631046004]: Not enough points(0) detected, stopping the robot
[INFO] [1489650306.633015814]: Not enough points(0) detected, stopping the robot
[INFO] [1489650307.666473100]: Not enough points(0) detected, stopping the robot
[INFO] [1489650308.668700141]: Not enough points(0) detected, stopping the robot
[INFO] [1489650309.668890368]: Not enough points(0) detected, stopping the robot
[INFO] [1489650310.670894125]: Not enough points(0) detected, stopping the robot
[INFO] [1489650311.738718761]: Not enough points(0) detected, stopping the robot
[INFO] [1489650312.738750552]: Not enough points(0) detected, stopping the robot
[INFO] [1489650313.740572702]: Not enough points(0) detected, stopping the robot
[INFO] [1489650314.741056975]: Not enough points(0) detected, stopping the robot
[INFO] [1489650315.743010189]: Not enough points(20) detected, stopping the robot
[INFO] [1489650316.776564916]: Not enough points(0) detected, stopping the robot
[INFO] [1489650316.876662436]: Centroid at 0.251242 0.147023 0.484000 with 4394 points
[INFO] [1489650317.878668378]: Centroid at -0.018614 -0.012538 0.651000 with 10022 points
[INFO] [1489650318.878695992]: Centroid at -0.114853 -0.055802 0.644000 with 20870 points
[INFO] [1489650319.880926621]: Centroid at -0.005633 -0.046356 0.583000 with 27740 points
[INFO] [1489650320.881171555]: Centroid at 0.039159 -0.045815 0.552000 with 31935 points
[INFO] [1489650321.883501304]: Centroid at -0.010385 -0.056388 0.617000 with 20585 points
[INFO] [1489650322.950670346]: Centroid at 0.000105 -0.050406 0.599000 with 24572 points
[INFO] [1489650323.951323699]: Centroid at 0.000688 -0.050407 0.600000 with 24593 points
[INFO] [1489650324.952694074]: Centroid at 0.000803 -0.050266 0.600000 with 24512 points
[INFO] [1489650325.952791577]: Centroid at 0.001065 -0.050412 0.600000 with 24542 points
[INFO] [1489650326.955043014]: Centroid at 0.001154 -0.050372 0.600000 with 24480 points
[INFO] [1489650327.956775824]: Centroid at 0.000900 -0.050333 0.600000 with 24478 points
```

Modify the parameters in the

`/home/xiaoqiang/Documents/ros/src/turtlebot_apps/turtlebot_follower/launch/freenect.launch` file on the Xiaoqiang host, similar to the following figure. Then you can control the following performance



```
freenect.launch — /run/user/1000/gvfs/sftp:host=192.168.0.112/home/xiaoqiang/Documents/ros/src/turtlebot_apps/turtlebot_follower/launch — Atom 17:37
launch
├── includes.launch
├── follower.launch
├── freenect.launch
├── kinect_frames.launch
├── kinect-follower.launch
├── xiaoqiang.rviz
└── xiaoqiangfollower.launch
freenect.launch
119 </include>
120
121 <!-- Real robot: Load turtlebot follower into the 3d sensors nodelet manager to avoid
122 <node pkg="nodelet" type="nodelet" name="turtlebot_follower"
123     args="load turtlebot_follower/TurtlebotFollower $(arg manager)">
124     <remap from="turtlebot_follower/cmd_vel" to="/cmd_vel"/>
125     <remap from="depth/points" to="$(arg depth)/points"/>
126     <param name="enabled" value="true" />
127     <param name="x_scale" value="10.0" />
128     <param name="z_scale" value="8.0" />
129     <param name="min_x" value="-0.4" />
130     <param name="max_x" value="0.4" />
131     <param name="min_y" value="-0.10" />
132     <param name="max_y" value="0.5" />
133     <param name="max_z" value="1.8" />
134     <param name="goal_z" value="0.6" />
135 </node>
136
137 </group> <!-- camera -->
```

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang ROS robot tutorial \(12\) display point cloud for kinect2](#)

xiaoqiang ROS robot tutorial (12) display point cloud for kinect2

[Xiaoqiang Homepage](#)

1.Start kinect v2 ROS driver

The Xiaoqiang platform outputs a 12V power supply (DC head with “kinect power supply” tag) for kinect power supply, and the kinect v2 needs to be inserted into the blue host USB 3.0 interface of the Xiaoqiang. Connect the Xiaoqiang host to the monitor and keyboard. Open a terminal on the Xiaoqiang host and enter the following command

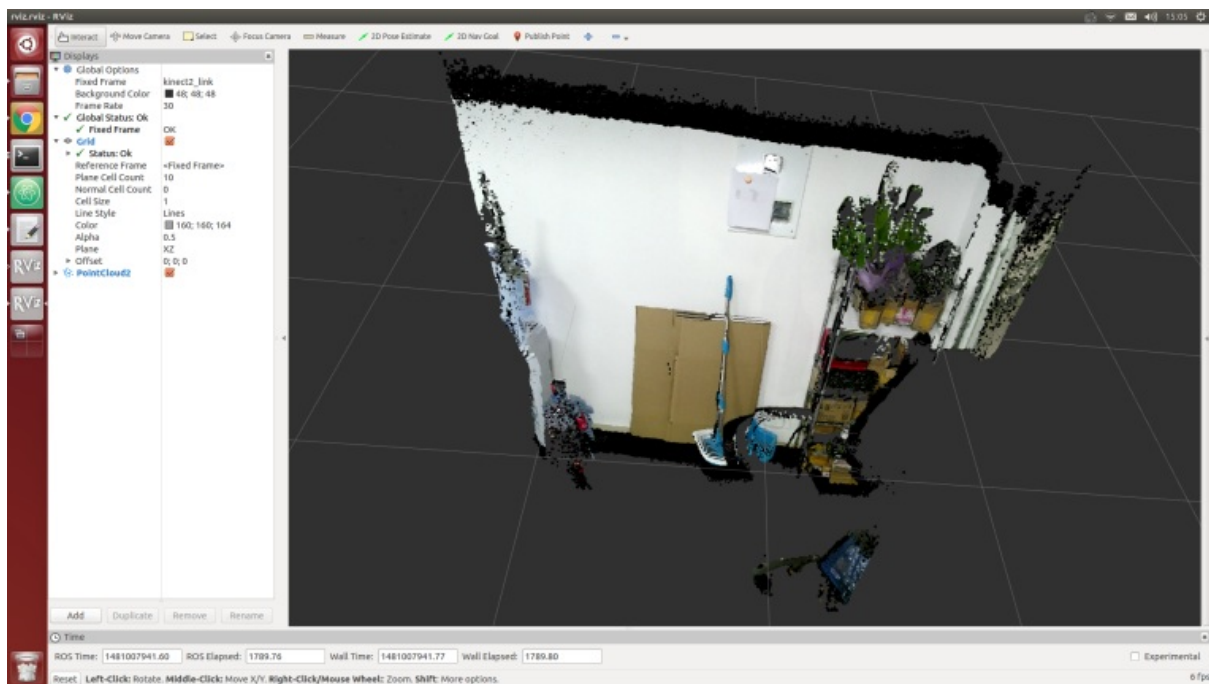
```
roslaunch kinect2_bridge kinect2-xyz.launch
```

2.Open a new terminal, start rviz

```
rviz
```

Open this rviz configuration file

`/home/xiaoqiang/Documents/ros/src/iai_kinect2/kinect2_bridge/launch/rviz.rviz` , [Click to download this file](#). If everything is normal, an interface similar to the one below can appear



[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(13\) rplidar A2 LiDAR useage and set udev rules for serial devices for xiaoqiang](#)
 - [rplidar A2 LiDAR useage and set udev rules for serial devices for xiaoqiang](#)

xiaoqiang tutorial (13) rplidar A2 LiDAR useage and set udev rules for serial devices for xiaoqiang

[Xiaoqiang Homepage](#)

rplidar A2 LiDAR useage and set udev rules for serial devices for xiaoqiang

The 1, 2, 3, and 4 steps of the tutorial in this section are only used to demonstrate the method of adding serial devices to Xiaoqiang. These rules are already added in Xiaoqiang system image. Xiaoqiang users should directly jump to step 5 to test the lidar after connecting the hardware.

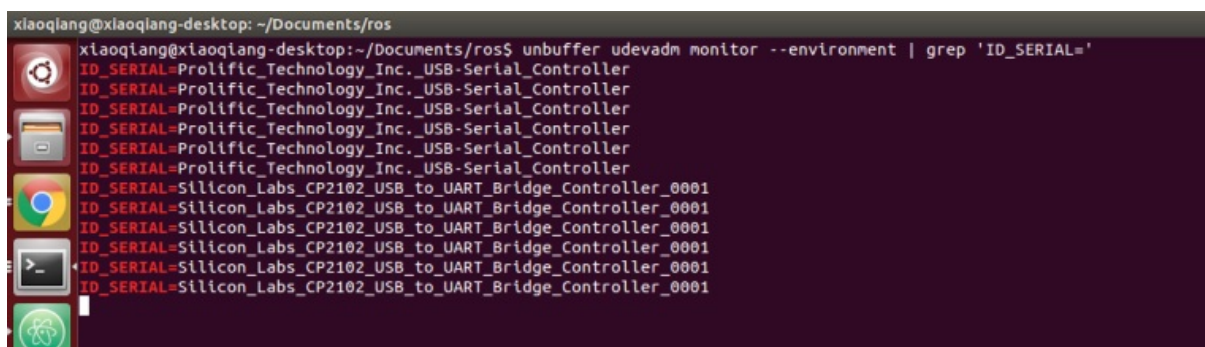
Xiaoqiang host and motor driver communicate through the serial port, In the actual development process, we may add serial peripherals to Xiaoqiang. This can lead to confusion over the serial number (ttyUSB*), causing anomalies in the xiaoqiang's ros driver and serial device. The following uses the rplidar A2 lidar as an example to demonstrate how to resolve serial port conflicts by modifying the device serial number specified in the udev file. The [source](#) of this method

1. Check the ID of each serial device

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

After the USB to serial cable of the platform is re-plugged, the terminal will print out the ID information of this device. For example, "Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller" in the following figure.

Then re-inserted the lidar's usb adapter into the host, and the terminal will also print the lidar's ID information. For example, "Prolific_Technology_Inc._USB-Serial_Controller_0001" in the following figure.



```
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. Please create a udev rule file based on the ID of the serial device you obtained. Specify platform communication USB to serial port as `ttyUSB001` and lidar as `ttyUSB002`.

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

Enter the following content and save it. Please replace the string following `ID_SERIAL` with the ID you obtained in step 1.

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

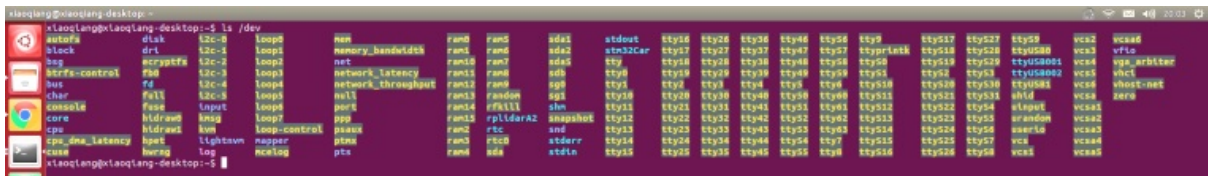
# This is old 11.10 style: IMPORT="usb_id --export %p"
IMPORT{builtin}="path_id"
ENV{ID_SERIAL}=="Prolific_Technology_Inc._USB-Serial_Controller" , SYMLINK="stm32P
latform" , SYMLINK+="ttyUSB001" , OWNER="xiaoqiang"
ENV{ID_SERIAL}=="Silicon_Labs_CP2102_USB_to_UART_Bridge_Controller_0001" , SYMLINK="rplida
rA2" , SYMLINK+="ttyUSB002" , OWNER="xiaoqiang"

LABEL="persistent_serial_end"
```

Update system udev rules

```
sudo udevadm control --reload
```

Re-plug all USB devices, now the platform communication USB to serial successfully identified as `tttyUSB001` , lidar is identified as `tttyUSB002` , regardless of the device insertion order and port.

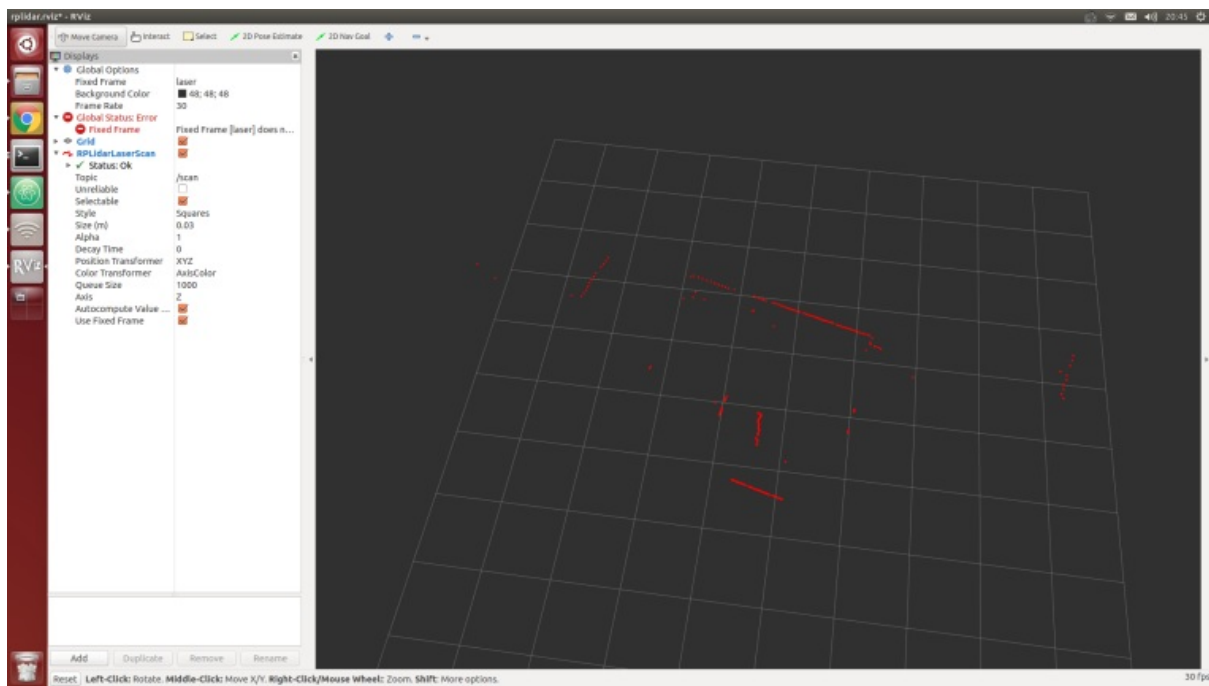


```
xiaoqiang@xiaoqiang-desktop:~$ ls /dev
autofs      disk        l2c-0       loop0       ram        ram0       ram5       sda1        sdbost     tty10       tty26       tty36       tty54       tty9
clock       dri         l2c-1       loop1       ram1       ram6       ram6       sda2        stbpcsr   tty11       tty27       tty37       tty55       tty91
klogd       fb          l2c-2       loop2       ram10      ram7       ram7       sda5        tty       tty14       tty28       tty38       tty56       tty92
librfc-control  fdc        l2c-3       loop3       ram11      ram8       ram8       sda6        tty0       tty19       tty29       tty39       tty57       tty93
log         fd          l2c-4       loop4       ram12      ram9       ram9       sda7        tty1       tty2         tty3         tty6         tty58       tty94
lirc        fsi         l2c-5       loop5       ram13      ram10      ram10      sda8        tty10      tty21       tty31       tty41       tty59       tty95
mknod       fuse       l2c-6       loop6       ram14      ram11      ram11      sda9        tty11      tty22       tty32       tty42       tty60       tty96
core        hidraw     loop7       loop7       ram15      ram12      ram12      rplidarA2  tty12      tty23       tty33       tty43       tty61       tty97
cpu_dma_latency  hpet      loop8       loop8       ram16      ram13      ram13      rplidarA2  tty13      tty24       tty34       tty44       tty62       tty98
cvs         hwrng     loop9       loop9       ram17      ram14      ram14      rplidarA2  tty14      tty25       tty35       tty45       tty63       tty99
dax         hvci       loop9       loop9       ram18      ram15      ram15      rplidarA2  tty15      tty26       tty36       tty46       tty64       tty100
dax         hvci       loop9       loop9       ram19      ram16      ram16      rplidarA2  tty16      tty27       tty37       tty47       tty65       tty101
dax         hvci       loop9       loop9       ram20      ram17      ram17      rplidarA2  tty17      tty28       tty38       tty48       tty66       tty102
dax         hvci       loop9       loop9       ram21      ram18      ram18      rplidarA2  tty18      tty29       tty39       tty49       tty67       tty103
dax         hvci       loop9       loop9       ram22      ram19      ram19      rplidarA2  tty19      tty30       tty40       tty50       tty68       tty104
dax         hvci       loop9       loop9       ram23      ram20      ram20      rplidarA2  tty20      tty31       tty41       tty51       tty69       tty105
dax         hvci       loop9       loop9       ram24      ram21      ram21      rplidarA2  tty21      tty32       tty42       tty52       tty70       tty106
dax         hvci       loop9       loop9       ram25      ram22      ram22      rplidarA2  tty22      tty33       tty43       tty53       tty71       tty107
dax         hvci       loop9       loop9       ram26      ram23      ram23      rplidarA2  tty23      tty34       tty44       tty54       tty72       tty108
dax         hvci       loop9       loop9       ram27      ram24      ram24      rplidarA2  tty24      tty35       tty45       tty55       tty73       tty109
dax         hvci       loop9       loop9       ram28      ram25      ram25      rplidarA2  tty25      tty36       tty46       tty56       tty74       tty110
dax         hvci       loop9       loop9       ram29      ram26      ram26      rplidarA2  tty26      tty37       tty47       tty57       tty75       tty111
dax         hvci       loop9       loop9       ram30      ram27      ram27      rplidarA2  tty27      tty38       tty48       tty58       tty76       tty112
dax         hvci       loop9       loop9       ram31      ram28      ram28      rplidarA2  tty28      tty39       tty49       tty59       tty77       tty113
dax         hvci       loop9       loop9       ram32      ram29      ram29      rplidarA2  tty29      tty40       tty50       tty60       tty78       tty114
dax         hvci       loop9       loop9       ram33      ram30      ram30      rplidarA2  tty30      tty41       tty51       tty61       tty79       tty115
dax         hvci       loop9       loop9       ram34      ram31      ram31      rplidarA2  tty31      tty42       tty52       tty62       tty80       tty116
dax         hvci       loop9       loop9       ram35      ram32      ram32      rplidarA2  tty32      tty43       tty53       tty63       tty81       tty117
dax         hvci       loop9       loop9       ram36      ram33      ram33      rplidarA2  tty33      tty44       tty54       tty64       tty82       tty118
dax         hvci       loop9       loop9       ram37      ram34      ram34      rplidarA2  tty34      tty45       tty55       tty65       tty83       tty119
dax         hvci       loop9       loop9       ram38      ram35      ram35      rplidarA2  tty35      tty46       tty56       tty66       tty84       tty120
dax         hvci       loop9       loop9       ram39      ram36      ram36      rplidarA2  tty36      tty47       tty57       tty67       tty85       tty121
dax         hvci       loop9       loop9       ram40      ram37      ram37      rplidarA2  tty37      tty48       tty58       tty68       tty86       tty122
dax         hvci       loop9       loop9       ram41      ram38      ram38      rplidarA2  tty38      tty49       tty59       tty69       tty87       tty123
dax         hvci       loop9       loop9       ram42      ram39      ram39      rplidarA2  tty39      tty50       tty60       tty70       tty88       tty124
dax         hvci       loop9       loop9       ram43      ram40      ram40      rplidarA2  tty40      tty51       tty61       tty71       tty89       tty125
dax         hvci       loop9       loop9       ram44      ram41      ram41      rplidarA2  tty41      tty52       tty62       tty72       tty90       tty126
dax         hvci       loop9       loop9       ram45      ram42      ram42      rplidarA2  tty42      tty53       tty63       tty73       tty91       tty127
dax         hvci       loop9       loop9       ram46      ram43      ram43      rplidarA2  tty43      tty54       tty64       tty74       tty92       tty128
dax         hvci       loop9       loop9       ram47      ram44      ram44      rplidarA2  tty44      tty55       tty65       tty75       tty93       tty129
dax         hvci       loop9       loop9       ram48      ram45      ram45      rplidarA2  tty45      tty56       tty66       tty76       tty94       tty130
dax         hvci       loop9       loop9       ram49      ram46      ram46      rplidarA2  tty46      tty57       tty67       tty77       tty95       tty131
dax         hvci       loop9       loop9       ram50      ram47      ram47      rplidarA2  tty47      tty58       tty68       tty78       tty96       tty132
dax         hvci       loop9       loop9       ram51      ram48      ram48      rplidarA2  tty48      tty59       tty69       tty79       tty97       tty133
dax         hvci       loop9       loop9       ram52      ram49      ram49      rplidarA2  tty49      tty60       tty70       tty80       tty98       tty134
dax         hvci       loop9       loop9       ram53      ram50      ram50      rplidarA2  tty50      tty61       tty71       tty81       tty99       tty135
dax         hvci       loop9       loop9       ram54      ram51      ram51      rplidarA2  tty51      tty62       tty72       tty82       tty100      tty136
dax         hvci       loop9       loop9       ram55      ram52      ram52      rplidarA2  tty52      tty63       tty73       tty83       tty101      tty137
dax         hvci       loop9       loop9       ram56      ram53      ram53      rplidarA2  tty53      tty64       tty74       tty84       tty102      tty138
dax         hvci       loop9       loop9       ram57      ram54      ram54      rplidarA2  tty54      tty65       tty75       tty85       tty103      tty139
dax         hvci       loop9       loop9       ram58      ram55      ram55      rplidarA2  tty55      tty66       tty76       tty86       tty104      tty140
dax         hvci       loop9       loop9       ram59      ram56      ram56      rplidarA2  tty56      tty67       tty77       tty87       tty105      tty141
dax         hvci       loop9       loop9       ram60      ram57      ram57      rplidarA2  tty57      tty68       tty78       tty88       tty106      tty142
dax         hvci       loop9       loop9       ram61      ram58      ram58      rplidarA2  tty58      tty69       tty79       tty89       tty107      tty143
dax         hvci       loop9       loop9       ram62      ram59      ram59      rplidarA2  tty59      tty70       tty80       tty90       tty108      tty144
dax         hvci       loop9       loop9       ram63      ram60      ram60      rplidarA2  tty60      tty71       tty81       tty91       tty109      tty145
dax         hvci       loop9       loop9       ram64      ram61      ram61      rplidarA2  tty61      tty72       tty82       tty92       tty110      tty146
dax         hvci       loop9       loop9       ram65      ram62      ram62      rplidarA2  tty62      tty73       tty83       tty93       tty111      tty147
dax         hvci       loop9       loop9       ram66      ram63      ram63      rplidarA2  tty63      tty74       tty84       tty94       tty112      tty148
dax         hvci       loop9       loop9       ram67      ram64      ram64      rplidarA2  tty64      tty75       tty85       tty95       tty113      tty149
dax         hvci       loop9       loop9       ram68      ram65      ram65      rplidarA2  tty65      tty76       tty86       tty96       tty114      tty150
dax         hvci       loop9       loop9       ram69      ram66      ram66      rplidarA2  tty66      tty77       tty87       tty97       tty115      tty151
dax         hvci       loop9       loop9       ram70      ram67      ram67      rplidarA2  tty67      tty78       tty88       tty98       tty116      tty152
dax         hvci       loop9       loop9       ram71      ram68      ram68      rplidarA2  tty68      tty79       tty89       tty99       tty117      tty153
dax         hvci       loop9       loop9       ram72      ram69      ram69      rplidarA2  tty69      tty80       tty90       tty100      tty118      tty154
dax         hvci       loop9       loop9       ram73      ram70      ram70      rplidarA2  tty70      tty81       tty91       tty101      tty119      tty155
dax         hvci       loop9       loop9       ram74      ram71      ram71      rplidarA2  tty71      tty82       tty92       tty102      tty120      tty156
dax         hvci       loop9       loop9       ram75      ram72      ram72      rplidarA2  tty72      tty83       tty93       tty103      tty121      tty157
dax         hvci       loop9       loop9       ram76      ram73      ram73      rplidarA2  tty73      tty84       tty94       tty104      tty122      tty158
dax         hvci       loop9       loop9       ram77      ram74      ram74      rplidarA2  tty74      tty85       tty95       tty105      tty123      tty159
dax         hvci       loop9       loop9       ram78      ram75      ram75      rplidarA2  tty75      tty86       tty96       tty106      tty124      tty160
dax         hvci       loop9       loop9       ram79      ram76      ram76      rplidarA2  tty76      tty87       tty97       tty107      tty125      tty161
dax         hvci       loop9       loop9       ram80      ram77      ram77      rplidarA2  tty77      tty88       tty98       tty108      tty126      tty162
dax         hvci       loop9       loop9       ram81      ram78      ram78      rplidarA2  tty78      tty89       tty99       tty109      tty127      tty163
dax         hvci       loop9       loop9       ram82      ram79      ram79      rplidarA2  tty79      tty90       tty100      tty110      tty128      tty164
dax         hvci       loop9       loop9       ram83      ram80      ram80      rplidarA2  tty80      tty91       tty101      tty111      tty129      tty165
dax         hvci       loop9       loop9       ram84      ram81      ram81      rplidarA2  tty81      tty92       tty102      tty112      tty130      tty166
dax         hvci       loop9       loop9       ram85      ram82      ram82      rplidarA2  tty82      tty93       tty103      tty113      tty131      tty167
dax         hvci       loop9       loop9       ram86      ram83      ram83      rplidarA2  tty83      tty94       tty104      tty114      tty132      tty168
dax         hvci       loop9       loop9       ram87      ram84      ram84      rplidarA2  tty84      tty95       tty105      tty115      tty133      tty169
dax         hvci       loop9       loop9       ram88      ram85      ram85      rplidarA2  tty85      tty96       tty106      tty116      tty134      tty170
dax         hvci       loop9       loop9       ram89      ram86      ram86      rplidarA2  tty86      tty97       tty107      tty117      tty135      tty171
dax         hvci       loop9       loop9       ram90      ram87      ram87      rplidarA2  tty87      tty98       tty108      tty118      tty136      tty172
dax         hvci       loop9       loop9       ram91      ram88      ram88      rplidarA2  tty88      tty99       tty109      tty119      tty137      tty173
dax         hvci       loop9       loop9       ram92      ram89      ram89      rplidarA2  tty89      tty100      tty110      tty120      tty138      tty174
dax         hvci       loop9       loop9       ram93      ram90      ram90      rplidarA2  tty90      tty101      tty111      tty121      tty139      tty175
dax         hvci       loop9       loop9       ram94      ram91      ram91      rplidarA2  tty91      tty102      tty112      tty122      tty140      tty176
dax         hvci       loop9       loop9       ram95      ram92      ram92      rplidarA2  tty92      tty103      tty113      tty123      tty141      tty177
dax         hvci       loop9       loop9       ram96      ram93      ram93      rplidarA2  tty93      tty104      tty114      tty124      tty142      tty178
dax         hvci       loop9       loop9       ram97      ram94      ram94      rplidarA2  tty94      tty105      tty115      tty125      tty143      tty179
dax         hvci       loop9       loop9       ram98      ram95      ram95      rplidarA2  tty95      tty106      tty116      tty126      tty144      tty180
dax         hvci       loop9       loop9       ram99      ram96      ram96      rplidarA2  tty96      tty107      tty117      tty127      tty145      tty181
dax         hvci       loop9       loop9       ram100      ram97      ram97      rplidarA2  tty97      tty108      tty118      tty128      tty146      tty182
dax         hvci       loop9       loop9       ram101      ram98      ram98      rplidarA2  tty98      tty109      tty119      tty129      tty147      tty183
dax         hvci       loop9       loop9       ram102      ram99      ram99      rplidarA2  tty99      tty110      tty120      tty130      tty148      tty184
dax         hvci       loop9       loop9       ram103      ram100      ram100      rplidarA2  tty100     tty111      tty121      tty131      tty149      tty185
dax         hvci       loop9       loop9       ram104      ram101      ram101      rplidarA2  tty101     tty112      tty122      tty132      tty150      tty186
dax         hvci       loop9       loop9       ram105      ram102      ram102      rplidarA2  tty102     tty113      tty123      tty133      tty151      tty187
dax         hvci       loop9       loop9       ram106      ram103      ram103      rplidarA2  tty103     tty114      tty124      tty134      tty152      tty188
dax         hvci       loop9       loop9       ram107      ram104      ram104      rplidarA2  tty104     tty115      tty125      tty135      tty153      tty189
dax         hvci       loop9       loop9       ram108      ram105      ram105      rplidarA2  tty105     tty116      tty126      tty136      tty154      tty190
dax         hvci       loop9       loop9       ram109      ram106      ram106      rplidarA2  tty106     tty117      tty127      tty137      tty155      tty191
dax         hvci       loop9       loop9       ram110      ram107      ram107      rplidarA2  tty107     tty118      tty128      tty138      tty156      tty192
dax         hvci       loop9       loop9       ram111      ram108      ram108      rplidarA2  tty108     tty119      tty129      tty139      tty157      tty193
dax         hvci       loop9       loop9       ram112      ram109      ram109      rplidarA2  tty109     tty120      tty130      tty140      tty158      tty194
dax         hvci       loop9       loop9       ram113      ram110      ram110      rplidarA2  tty110     tty121      tty131      tty141      tty159      tty195
dax         hvci       loop9       loop9       ram114      ram111      ram111      rplidarA2  tty111     tty122      tty132      tty142      tty160      tty196
dax         hvci       loop9       loop9       ram115      ram112      ram112      rplidarA2  tty112     tty123      tty133      tty143      tty161      tty197
dax         hvci       loop9       loop9       ram116      ram113      ram113      rplidarA2  tty113     tty124      tty134      tty144      tty162      tty198
dax         hvci       loop9       loop9       ram117      ram114      ram114      rplidarA2  tty114     tty125      tty135      tty145      tty163      tty199
dax         hvci       loop9       loop9       ram118      ram115      ram115      rplidarA2  tty115     tty126      tty136      tty146      tty164      tty200
dax         hvci       loop9       loop9       ram119      ram116      ram116      rplidarA2  tty116     tty127      tty137      tty147      tty165      tty201
dax         hvci       loop9       loop9       ram120      ram117      ram117      rplidarA2  tty117     tty128      tty138      tty148      tty166      tty202
dax         hvci       loop9       loop9       ram121      ram118      ram118      rplidarA2  tty118     tty129      tty139      tty149      tty167      tty203
dax         hvci       loop9       loop9       ram122      ram119      ram119      rplidarA2  tty119     tty130      tty140      tty150      tty168      tty204
dax         hvci       loop9       loop9       ram123      ram120      ram120      rplidarA2  tty120     tty131      tty141      tty151      tty169      tty205
dax         hvci       loop9       loop9       ram124      ram121      ram121      rplidarA2  tty121     tty132      tty142      tty152      tty170      tty206
dax         hvci       loop9       loop9       ram125      ram122      ram122      rplidarA2  tty122     tty133      tty143      tty153      tty171      tty207
dax         hvci       loop9       loop9       ram126      ram123      ram123      rplidarA2  tty123     tty134      tty144      tty154      tty172      tty208
dax         hvci       loop9       loop9       ram127      ram124      ram124      rplidarA2  tty124     tty135      tty145      tty155      tty173      tty209
dax         hvci       loop9       loop9       ram128      ram125      ram125      rplidarA2  tty125     tty136      tty146      tty156      tty174      tty210
dax         hvci       loop9       loop9       ram129      ram126      ram126      rplidarA2  tty126     tty137      tty147      tty157      tty175      tty211
dax         hvci       loop9       loop9       ram130      ram127      ram127      rplidarA2  tty127     tty138      tty148      tty158      tty176      tty212
dax         hvci       loop9       loop9       ram131      ram128      ram128      rplidarA2  tty128     tty139      tty149      tty159      tty177      tty213
dax         hvci       loop9       loop9       ram132      ram129      ram129      rplidarA2  tty129     tty140      tty150      tty160      tty178      tty214
dax         hvci       loop9       loop9       ram133      ram130      ram130      rplidarA2  tty130     tty141      tty151      tty161      tty179      tty215
dax         hvci       loop9       loop9       ram134      ram131      ram131      rplidarA2  tty131     tty142      tty152      tty162      tty180      tty216
dax         hvci       loop9       loop9       ram135      ram132      ram132      rplidarA2  tty132     tty143      tty153      tty163      tty181      tty217
dax         hvci       loop9       loop9       ram136      ram133      ram133      rplidarA2  tty133     tty144      tty154      tty164      tty182      tty218
dax         hvci       loop9       loop9       ram137      ram134      ram134      rplidarA2  tty134     tty145      tty155      tty165      tty183      tty219
dax         hvci       loop9       loop9       ram138      ram135      ram135      rplidarA2  tty135     tty146      tty156      tty166      tty184      tty220
dax         hvci       loop9       loop9       ram139      ram136      ram136      rplidarA2  tty136     tty147      tty157      tty167      tty185      tty221
dax         hvci       loop9       loop9       ram140      ram137      ram137      rplidarA2  tty137     tty148      tty158      tty168      tty186      tty222
dax         hvci       loop9       loop9       ram141      ram138      ram138      rplidarA2  tty138     tty149      tty159      tty169      tty187      tty223
dax         hvci       loop9       loop9       ram142      ram139      ram139      rplidarA2  tty139     tty150      tty160      tty170      tty188      tty224
dax         hvci       loop9       loop9       ram143      ram140      ram140      rplidarA2  tty140     tty151      tty161      tty171      tty189      tty225
dax         hvci       loop9       loop9       ram144      ram141      ram141      rplidarA2  tty141     tty152      tty162      tty172      tty190      tty226
dax         hvci       loop9       loop9       ram145      ram142      ram142      rplidarA2  tty142     tty153      tty163      tty173      tty191      tty227
dax         hvci       loop9       loop9       ram146      ram143      ram143      rplidarA2  tty143     tty154      tty164      tty174      tty192      tty228
dax         hvci       loop9       loop9       ram147      ram144      ram144      rplidarA2  tty144     tty155      tty165      tty175      tty193      tty229
dax         hvci       loop9       loop9       ram148      ram145      ram145      rplidarA2  tty145     tty156      tty166      tty176      tty194      tty230
dax         hvci       loop9       loop9       ram149      ram146      ram146      rplidarA2  tty146     tty157      tty167      tty177      tty195      tty231
dax         hvci       loop9       loop9       ram150      ram147      ram147      rplidarA2  tty147     tty158      tty168      tty178      tty196      tty232
dax         hvci       loop9       loop9       ram151      ram148      ram148      rplidarA2  tty148     tty159      tty169      tty179      tty197      tty233
dax         hvci       loop9       loop9       ram152      ram149      ram149      rplidarA2  tty149     tty160      tty170      tty180      tty198      tty234
dax         hvci       loop9       loop9       ram153      ram150      ram150      rplidarA2  tty150     tty161      tty171      tty181      tty199      tty235
dax         hvci       loop9       loop9       ram154      ram151      ram151      rplidarA2  tty151     tty162      tty172      tty182      tty200      tty236
dax         hvci       loop9       loop9       ram155      ram152      ram152      rplidarA2  tty152     tty163      tty173      tty183      tty201      tty237
dax         hvci       loop9       loop9       ram156      ram153      ram153      rplidarA2  tty153     tty164      tty174      tty184      tty202      tty238
dax         hvci       loop9       loop9       ram157      ram154      ram154      rplidarA2  tty154     tty165      tty175      tty185      tty203      tty239
dax         hvci       loop9       loop9       ram158      ram155      ram155      rplidarA2  tty155     tty166      tty176      tty186      tty204      tty240
dax         hvci       loop9       loop9       ram159      ram156      ram156      rplidarA2  tty156     tty167      tty177      tty187      tty205      tty241
dax         hvci       loop9       loop9       ram160      ram157      ram157      rplidarA2  tty157     tty168      tty178      tty188      tty206      tty242
dax         hvci       loop9       loop9       ram161      ram158      ram158      rplidarA2  tty158     tty169      tty179      tty189      tty207      tty243
dax         hvci       loop9       loop9       ram162      ram159      ram159      rplidarA2  tty159     tty170      tty180      tty190      tty208      tty244
dax         hvci       loop9       loop9       ram163      ram160      ram160      rplidarA2  tty160     tty171      tty181      tty191      tty209      tty245
dax         hvci       loop9       loop9       ram164      ram161      ram161      rplidarA2  tty161     tty172      tty182      tty192      tty210      tty246
dax         hvci       loop9       loop9       ram165      ram162      ram162      rplidarA2  tty162     tty173      tty183      tty193      tty211      tty247
dax         hvci       loop9       loop9       ram166      ram163      ram163      rplidarA2  tty163     tty174      tty184      tty194      tty212      tty248
dax         hvci       loop9       loop9       ram167      ram164      ram164      rplidarA2  tty164     tty175      tty185      tty195      tty213      tty249
dax         hvci       loop9       loop9       ram168      ram165      ram165      rplidarA2  tty165     tty176      tty186      tty196      tty214      tty250
dax         hvci       loop9       loop9       ram169      ram166      ram166      rplidarA2  tty166     tty177      tty187      tty197      tty215      tty251
dax         hvci       loop9       loop9       ram170      ram167      ram167      rplidarA2  tty167     tty178      tty188      tty198      tty216      tty252
dax         hvci       loop9       loop9       ram171      ram168      ram168      rplidarA2  tty168     tty179      tty189      tty199      tty217      tty253
dax         hvci       loop9       loop9       ram172      ram169      ram169      rplidarA2  tty169     tty180      tty190      tty200      tty218      tty254
dax         hvci       loop9       loop9       ram173      ram170      ram170      rplidarA2  tty170     tty181      tty191      tty201      tty219      tty255
dax         hvci       loop9       loop9       ram174      ram171      ram171      rplidarA2  tty171     tty182      tty192      tty202      tty220      tty256
dax         hvci       loop9       loop9       ram175      ram172      ram172      rplidarA2  tty172     tty183      tty193      tty203      tty221      tty257
dax         hvci       loop9       loop9       ram176      ram173      ram173      rplidarA2  tty173     tty184      tty194      tty204      tty222      tty258
dax         hvci       loop9       loop9       ram177      ram174      ram174      rplidarA2  tty174     tty185      tty195      tty205      tty223      tty259
dax         hvci       loop9       loop9       ram178      ram175      ram175      rplidarA2  tty175     tty186      tty196      tty206      tty224      tty260
dax         hvci       loop9       loop9       ram179      ram176      ram176      rplidarA2  tty176     tty187      tty197      tty207      tty225      tty261
dax         hvci       loop9       loop9       ram180      ram177      ram177      rplidarA2  tty177     tty188      tty198      tty208      tty226      tty262
dax         hvci       loop9       loop9       ram181      ram178      ram178      rplidarA2  tty178     tty189      tty199      tty209      tty227      tty263
dax         hvci       loop9       loop9       ram182      ram179      ram179      rplidarA2  tty179     tty190      tty200      tty210      tty228      tty264
dax         hvci       loop9       loop9       ram183      ram180      ram180      rplidarA2  tty180     tty191      tty201      tty211      tty229      tty265
dax         hvci       loop9       loop9       ram184      ram181      ram181      rplidarA2  tty181     tty192      tty202      tty212      tty230      tty266
dax         hvci       loop9       loop9       ram185      ram182      ram182      rplidarA2  tty182     tty193      tty203      tty213      tty231      tty267
dax         hvci       loop9       loop9       ram186      ram183      ram183      rplidarA2  tty183     tty194      tty204      tty214      tty232      tty268
dax         hvci       loop9       loop9       ram187      ram184      ram184      rplidarA2  tty184     tty195      tty205      tty215      tty233      tty269
dax         hvci       loop9       loop9       ram188      ram185      ram185      rplidarA2  tty185     tty196      tty206      tty216      tty234      tty270
dax         hvci       loop9       loop9       ram189      ram186      ram186      rplidarA2  tty186     tty197      tty207      tty217      tty235      tty271
dax         hvci       loop9       loop9       ram190      ram187      ram187      rplidarA2  tty187     tty198      tty208      tty218      tty236      tty272
dax         hvci       loop9       loop9       ram191      ram188      ram188      rplidarA2  tty188     tty199      tty209      tty219      tty237      tty273
dax         hvci       loop9       loop9       ram192      ram189      ram189      rplidarA2  tty189     tty200      tty210      tty220      tty238      tty274
dax         hvci       loop9       loop9       ram193      ram190      ram190      rplidarA2  tty190     tty201      tty211      tty221      tty239      tty275
dax         hvci       loop9       loop9       ram194      ram191      ram191      rplidarA2  tty191     tty202      tty212      tty222      tty240      tty276
dax         hvci       loop9       loop9       ram195      ram192      ram192      rplidarA2  tty192     tty203      tty213      tty223      tty241      tty277
dax         hvci       loop9       loop9       ram196      ram193      ram193      rplidarA2  tty193     tty204      tty214      tty224      tty242      tty278
dax         hvci       loop9       loop9       ram197      ram194      ram194      rplidarA2  tty194     tty205      tty215      tty225      tty243      tty279
dax         hvci       loop9       loop9       ram198      ram195      ram195      rplidarA2  tty195     tty206      tty216      tty226      tty244      tty280
dax         hvci       loop9       loop9       ram199      ram196      ram196      rplidarA2  tty196     tty207      tty217      tty227      tty245      tty281
dax         hvci       loop9       loop9       ram200      ram197      ram197      rplidarA2  tty197     tty208      tty218      tty228      tty246      tty282
dax         hvci       loop9       loop9       ram201      ram198      ram198      rplidarA2  tty198     tty209      tty219      tty229      tty247      tty283
dax         hvci       loop9       loop9       ram202      ram199      ram199      rplidarA2  tty199     tty210      tty220      tty230      tty248      tty284
dax         hvci       loop9       loop9       ram203      ram200      ram200      rplidarA2  tty200     tty211      tty221      tty231      tty249     
```

```
rplidar.launch (~/Documents/ros/src/rplidar_ros/launch) - gedit
rplidar.launch x test_rplidar.launch x
<launch>
  <node name="rplidarNode" pkg="rplidar_ros" type="rplidarNode" output="screen">
    <param name="serial_port" type="string" value="/dev/ttyUSB002"/>
    <param name="serial_baudrate" type="int" value="115200"/>
    <param name="frame_id" type="string" value="laser"/>
    <param name="inverted" type="bool" value="false"/>
    <param name="angle_compensate" type="bool" value="true"/>
  </node>
</launch>
```

1. After restarting Xiaoqiang, it is now possible to use the lidar and xiaoqiang normally at the same time. For example, run the following command to test the lidar.

```
roslaunch rplidar_ros view_rplidar.launch
```



[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(14\) using rplidar A2 with gmapping](#)
 - [Using rplidar A2 with gmapping](#)
 - [1.Start the gmapping node](#)
 - [2.Remote control Xiaoqiang and start to build a map](#)
 - [3.Save the map](#)

xiaoqiang tutorial (14) using rplidar A2 with gmapping

[Xiaoqiang Homepage](#)

Using rplidar A2 with gmapping

1.Start the gmapping node

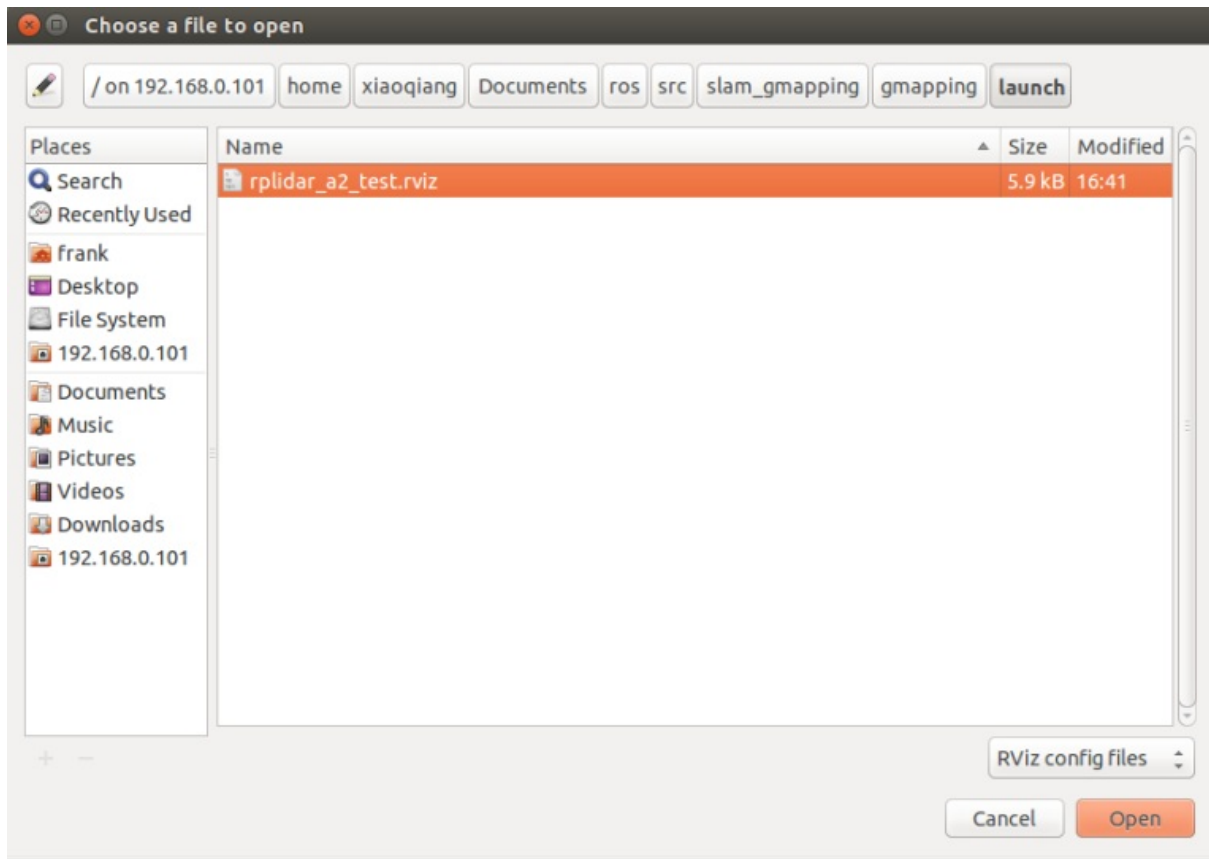
Ensure that the lidar is properly installed, and launch the gmapping launch file after ssh login the host.

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

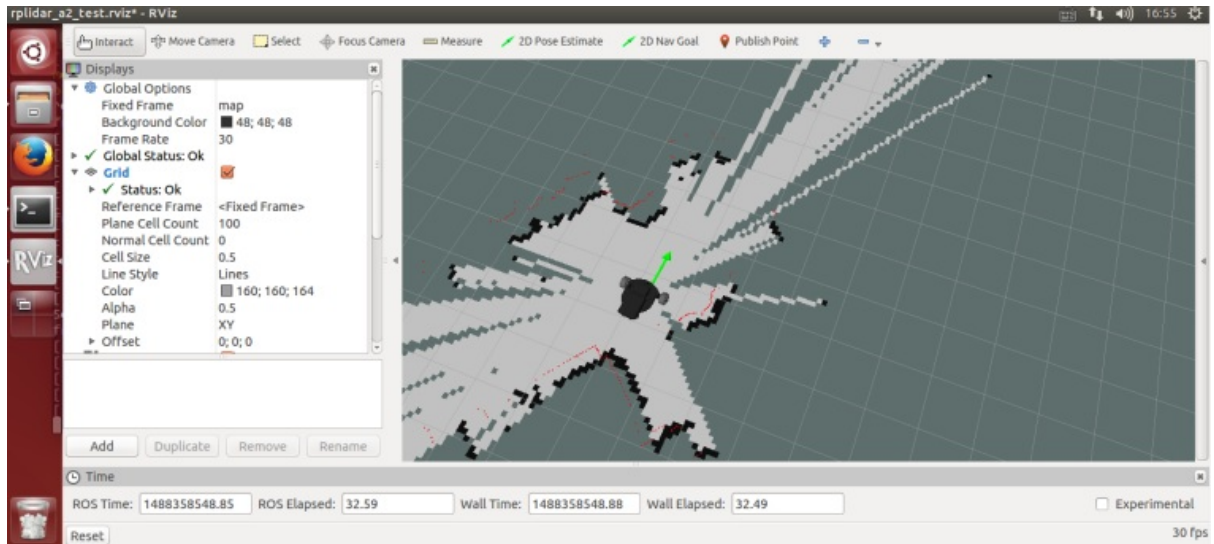
Open rviz in the local virtual machine, select the `slam_gmapping/gmapping/launch/rplidar_a2_test.rviz` configuration file and open it in Xiaoqiang ROS directory.

Note: Some systems can't do this. You can copy that file directly to your local computer and open it locally.

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

Wait a few seconds, normal results similar to the following figure will appear.



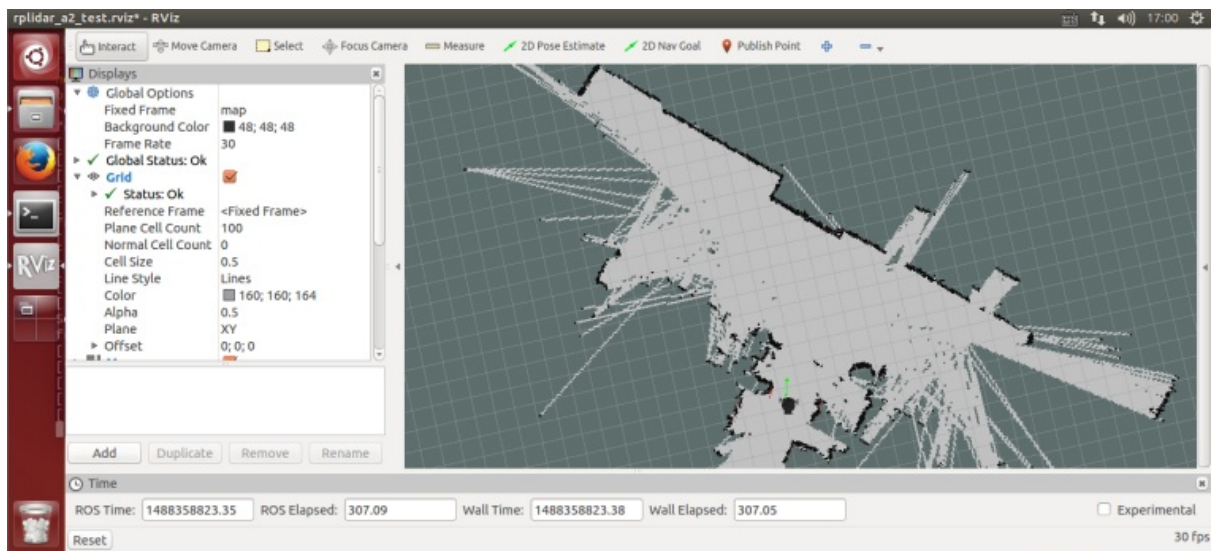
2.Remote control Xiaoqiang and start to build a map

The first way: use the windows remote control client, refer to this [post](#)

The second way:using the ssh remote control

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

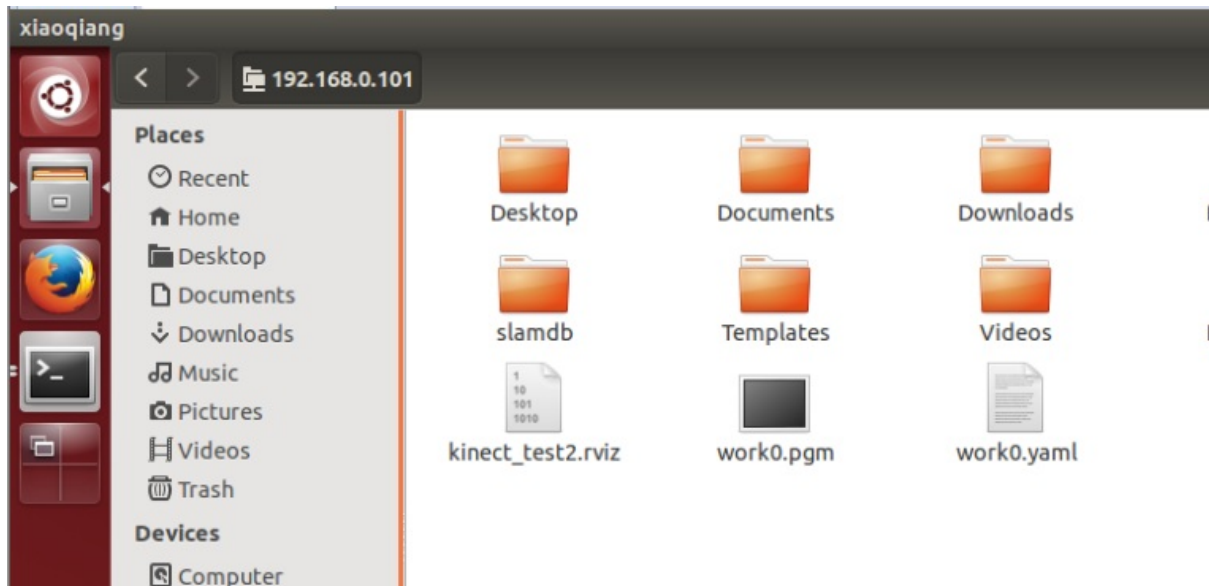
The third way: use mobile app, refer to this post [Xiaoqiang mobile remote control app for Android](#)



3. Save the map

ssh login Xiaoqiang, in the Xiaoqiang home directory save the map as the beginning of work0 file.

```
ssh xiaoqiang@192.168.XXX.XXX
roslaunch map_server map_saver -f work0
```



[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(15\) AMCL navigation test](#)
 - [AMCL Navigation Test](#)
 - [1 preparations](#)
 - [2.Start navigation node](#)
 - [3 Start rviz](#)
 - [4.Start navigation test](#)
 - [5. Set up Xiaoqiang's 2D Nav Goal and observe the movement of Xiaoqiang.](#)

xiaoqiang tutorial (15) AMCL navigation test

[Xiaoqiang Homepage](#)

AMCL Navigation Test

The following will demonstrate the AMCL navigation operation using rplidar a2 as the scan input and the map file created in [Tutorial 14](#) as the global map.

1 preparations

Install and upgrade `nav_test` , `laser_filters` package first.

1.1 ssh login Xiaoqiang host and cd to Xiaoqiang's ros workspace

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

1.2 Update upgrade package

```
cd laser_filters
git stash
git pull
cd ..
cd nav_test
git stash
git pull
cd ..
cd ..
catkin_make
```

1.3 Update the Xiaoqiang hosts file and the hosts file of the local virtual machine so that Xiaoqiang and the local virtual machine can communicate with each other. Refer to sections 1.A and 1.B in [Tutorial 13](#).

2.Start navigation node

First, copy the two map files created in [Tutorial 14](#) to

`/home/xiaoqiang/Documents/ros/src/nav_test/maps/` , overwrite the files with the same name.

```
roslaunch nav_test xiaoqiang_a2_demo_amcl.launch
```

Normally there will be similar results in the figure below, while the lidar starts to rotate

```

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/pi
x
[ INFO] [1488440695.803708785]: Received a 1984 X 1984 map at 0.050000 m/pix
[ INFO] [1488440695.808433410]: Using plugin "obstacle_layer"
[ INFO] [1488440695.811323486]: Subscribed to Topics: scan
[ INFO] [1488440695.835866250]: Using plugin "inflation_layer"
[ERROR] [1488440695.851557790]: You must specify at least three points for the r
obot footprint, reverting to previous footprint.
[ INFO] [1488440695.910781620]: Using plugin "static_layer"
[ INFO] [1488440695.936803446]: Requesting the map...
[ INFO] [1488440695.940642889]: Resizing static layer to 1984 X 1984 at 0.050000
m/pix
[ INFO] [1488440696.039063201]: Received a 1984 X 1984 map at 0.050000 m/pix
[ INFO] [1488440696.043339673]: Using plugin "obstacle_layer"
[ INFO] [1488440696.045275723]: Subscribed to Topics: scan
[ INFO] [1488440696.065123464]: Using plugin "inflation_layer"
[ERROR] [1488440696.077350828]: You must specify at least three points for the r
obot footprint, reverting to previous footprint.
[ INFO] [1488440696.129140042]: ODOM SET1!

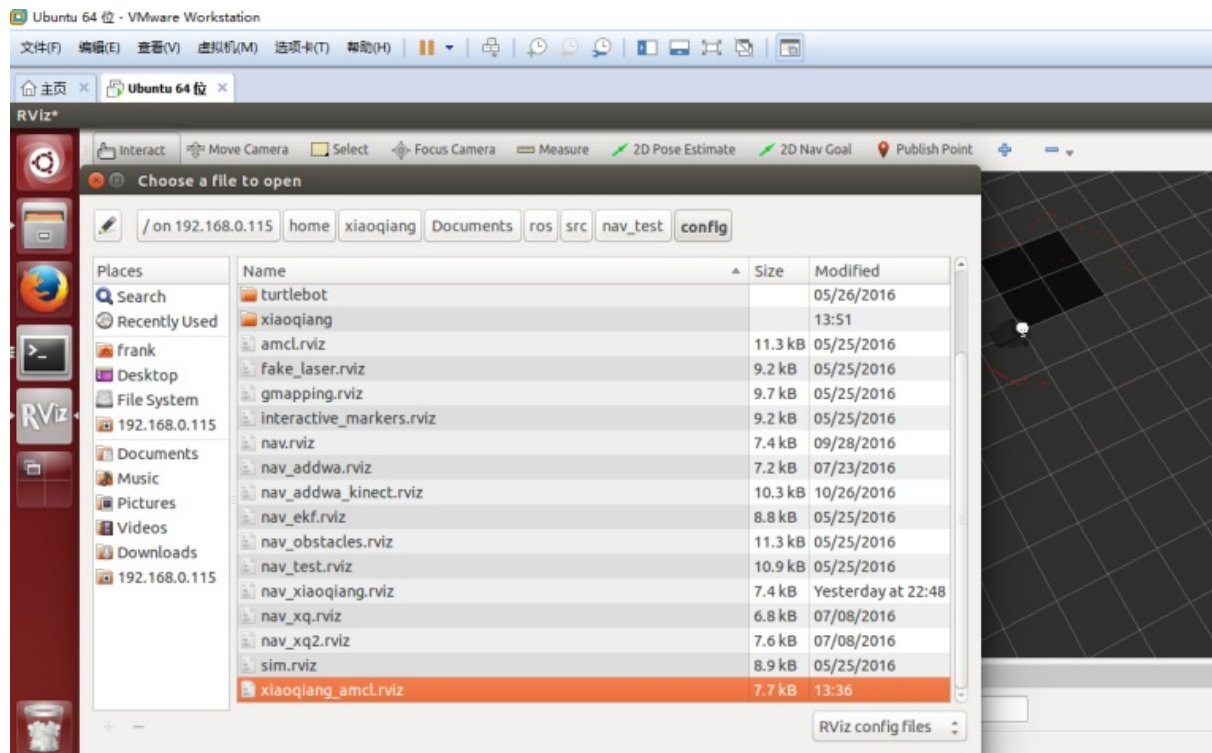
[ INFO] [1488440696.129194840]: Created local_planner dwa_local_planner/DWAPlann
erROS
[ INFO] [1488440696.132244917]: Sim period is set to 0.10
[ INFO] [1488440696.458625520]: Recovery behavior will clear layer obstacles
[ INFO] [1488440696.483303827]: Recovery behavior will clear layer obstacles

```

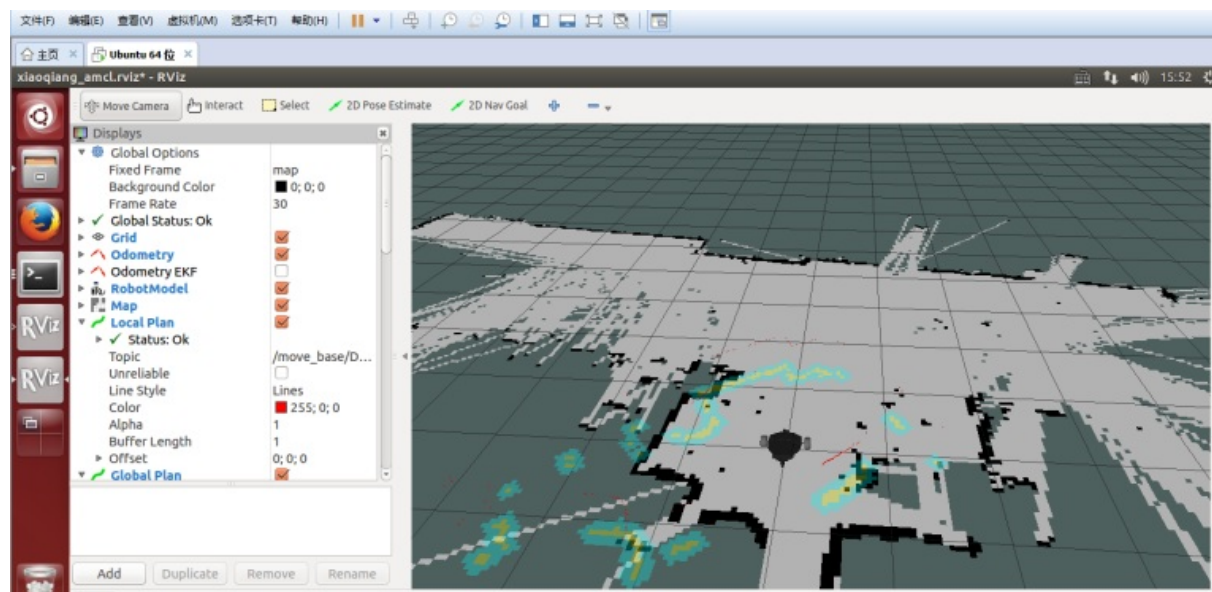
3 Start rviz

3.1 Start rviz in the local virtual, select the open `nav_test/config/xiaoqiang_amcl.rviz` configuration file in the work directory of Xiaoqiang ros

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

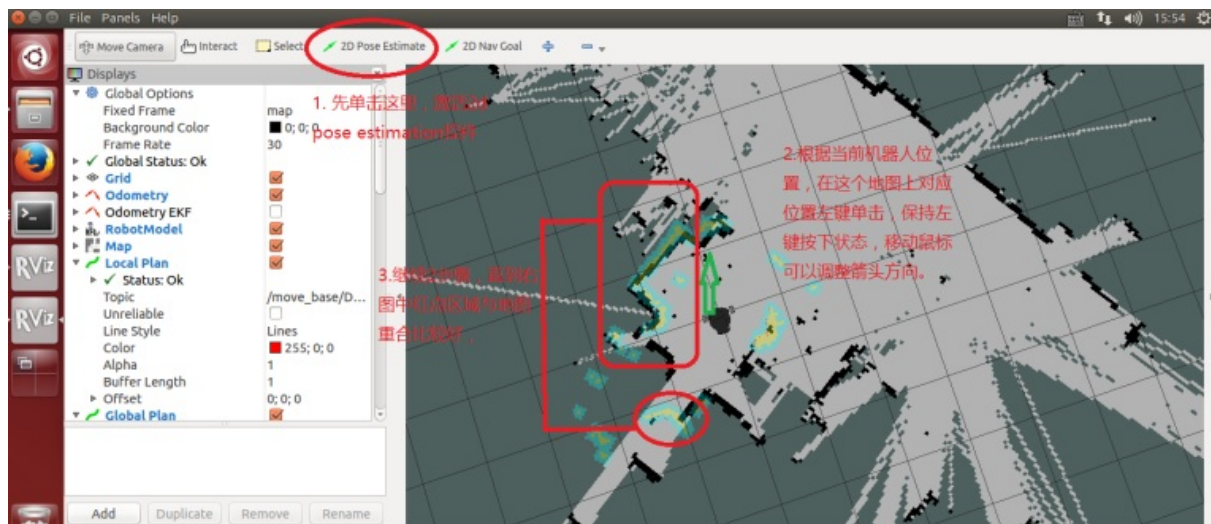


3.2 After waiting for a few seconds, rviz will normally have a screen similar to the one below.

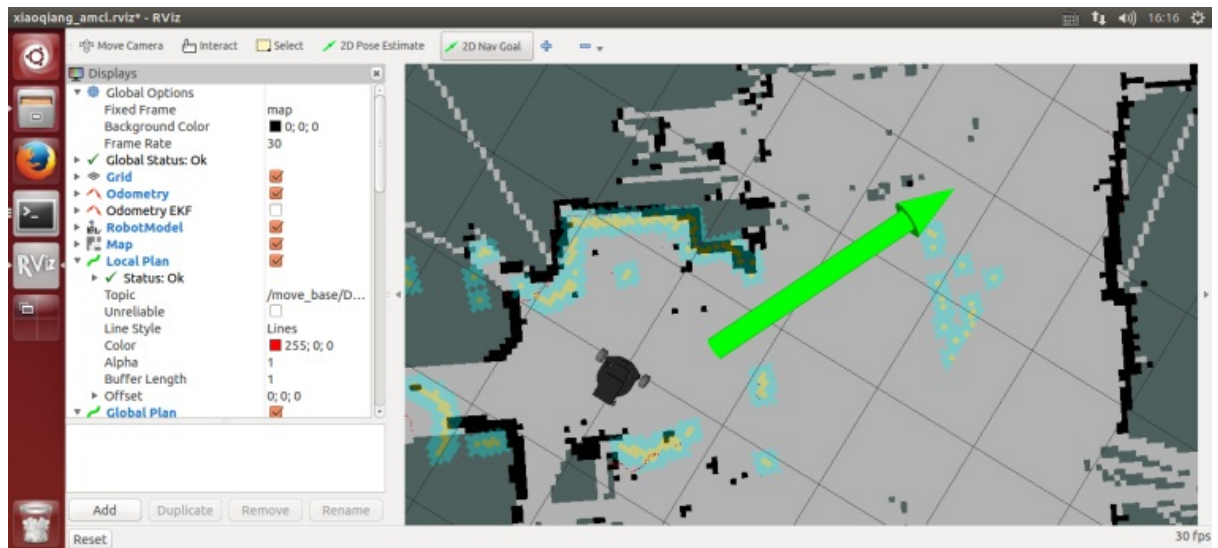


4. Start navigation test

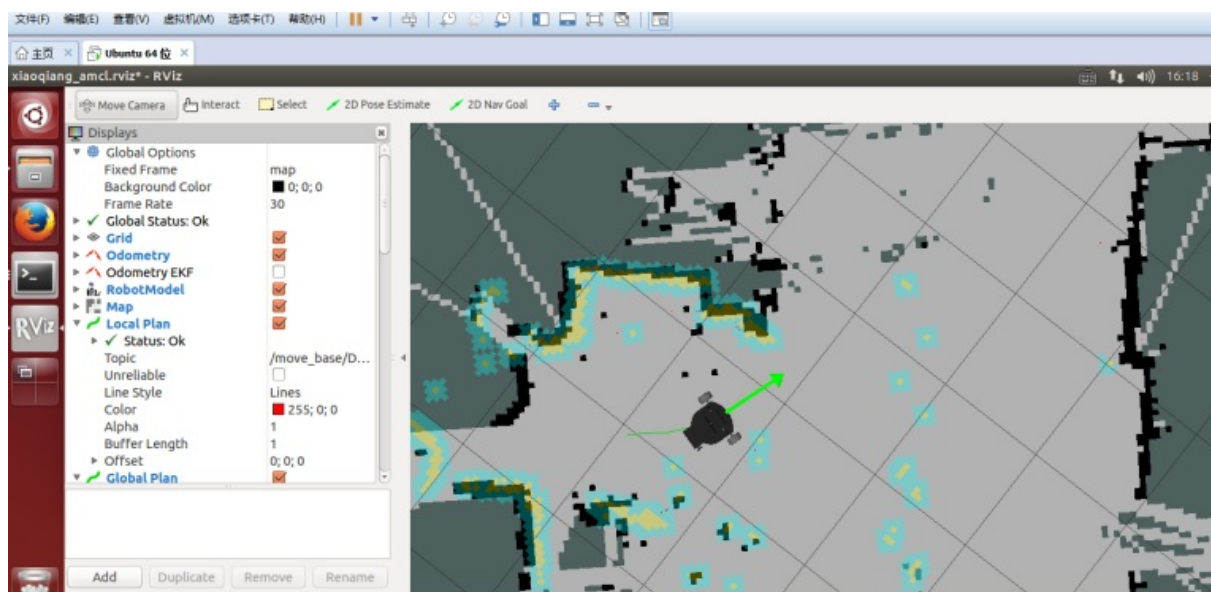
4.1 In rviz, use 2D Pose Estimation to set the position of the initial pose of the robot in the map. Because the AMCL algorithm needs a more accurate initial value, the actual position of the robot in the map can be further matched by the current lidar scanning dot matrix.



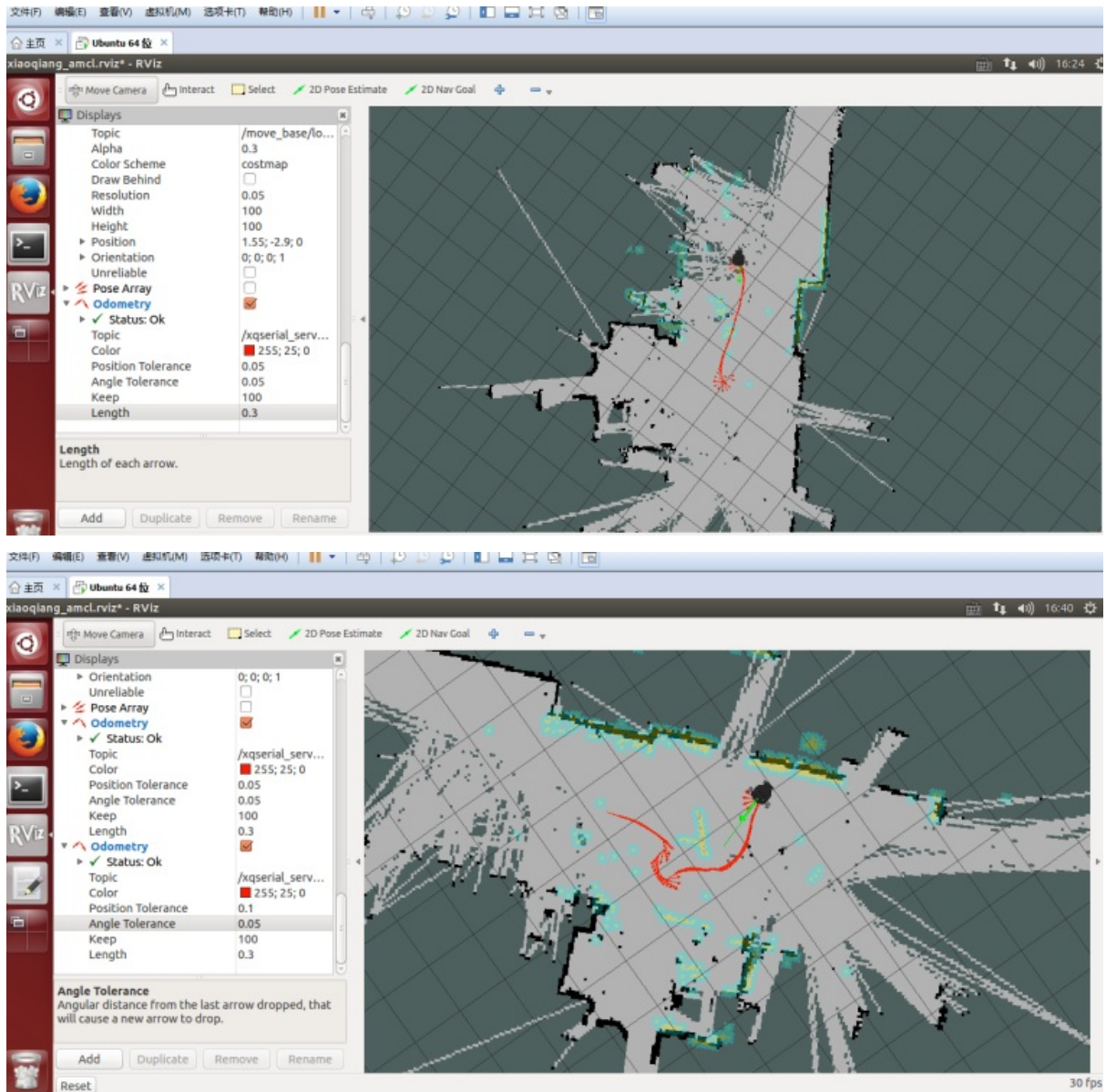
4.2 Using 2D Nav Goal in rviz to release targets to Xiaoqiang



4.3 Xiaoqiang began to move autonomously to the designated location



5. Set up Xiaoqiang's 2D Nav Goal and observe the movement of Xiaoqiang.



[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(16\) large-scale lidar slam and real-time loop closure test](#)
 - [Large-scale lidar slam and real-time loop closure test](#)
 - Preparation :

xiaoqiang tutorial (16) large-scale lidar slam and real-time loop closure test

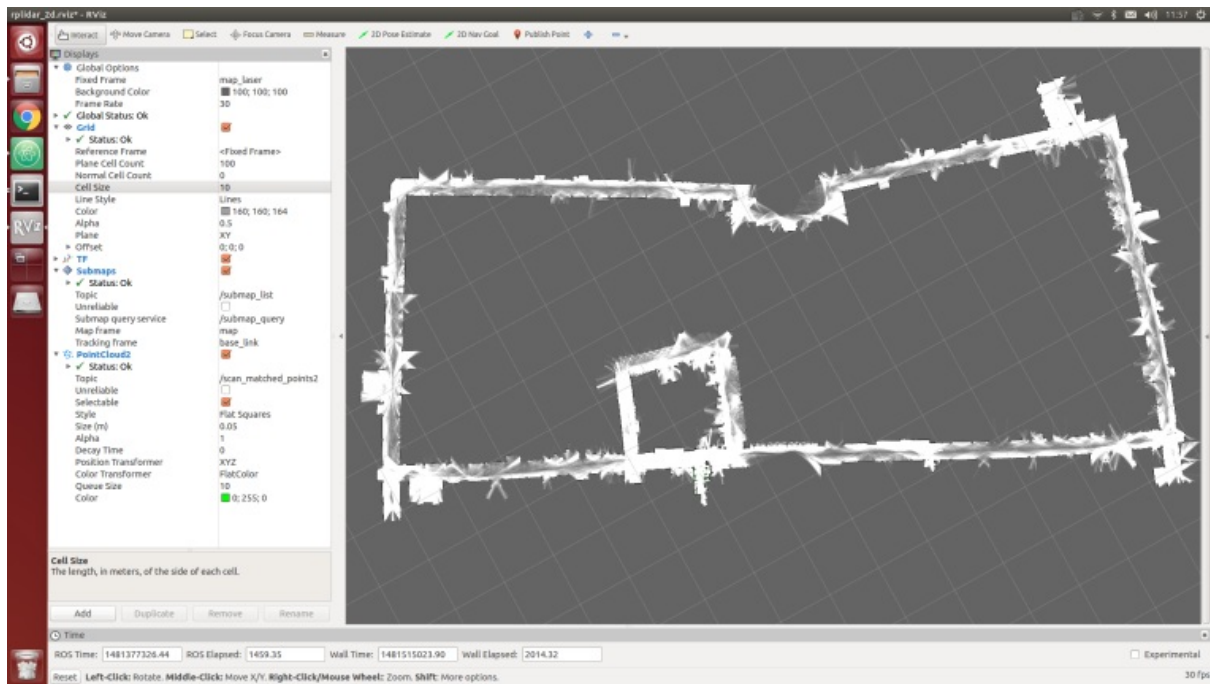
[Xiaoqiang Homepage](#)

Large-scale lidar slam and real-time loop closure test

With Google's Cartographer and slamtec's lidar, we can try to create a floor plan for a large building. Take a look at our demo and click to watch the video.



In this demo, Xiaoqiang actually operates in a 5,000-square-meter office building corridor. There are a large number of glass walls on both sides of the corridor. There is a large area of open space in the center of the building. With rplidar's range of only 6 meters, so the following figure The final result is still relatively good (only use lidar, IMU and odometer are not turned on, the large loop path is still closed successfully)



The idea of this article: Because it is a large-scale mapping, wifi network coverage is a problem, so we use the Bluetooth joystick to remotely control the robot movement. During the remote control, the lidar data was recorded through the rosbag. The joystick controlled robot ran a circle within the area and finally play the bag for map creation.

Note: All the following operations are completed on Xiaoqiang host ubuntu

Preparation:

1.Installing rplidar driver

If you are using xiaoqiang system image, then the driver was already installed, you can skip this step.

For the instructions to install rplidar driver, please refer to this [post](#)

2.Install PS3 handle driver

If you are using xiaoqiang system image, then the driver was already installed, you can skip this step.

For the instructions to install joystick driver, please refer to this [post](#)

3.Install cartographer_ros

Please refer to this [installation tutorial](#)

Steps:

1.Open a new window to start rplidar

```
roslaunch rplidar_ros rplidar.launch
```

2. Open two windows to start PS3 controller remote control program, press the handle connection key to connect the platform.

First window

```
sudo bash
roslaunch ps3joy ps3joyfake_node.py
```

The second window

```
roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
```

3. Opens a new window to start the rosbag recording process and starts recording lidar data/scan

```
rosbag record /scan
```

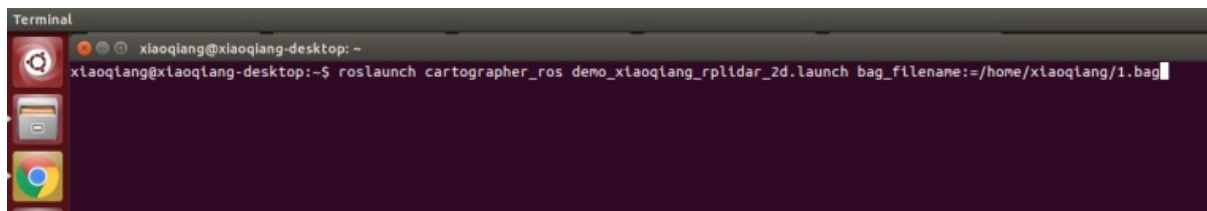
4. Use the handle to remotely control the movement of the platform, circle around the building area, and it can also be used for multiple rotations.

5. Bag recording is complete, close the above 1,2,3 window

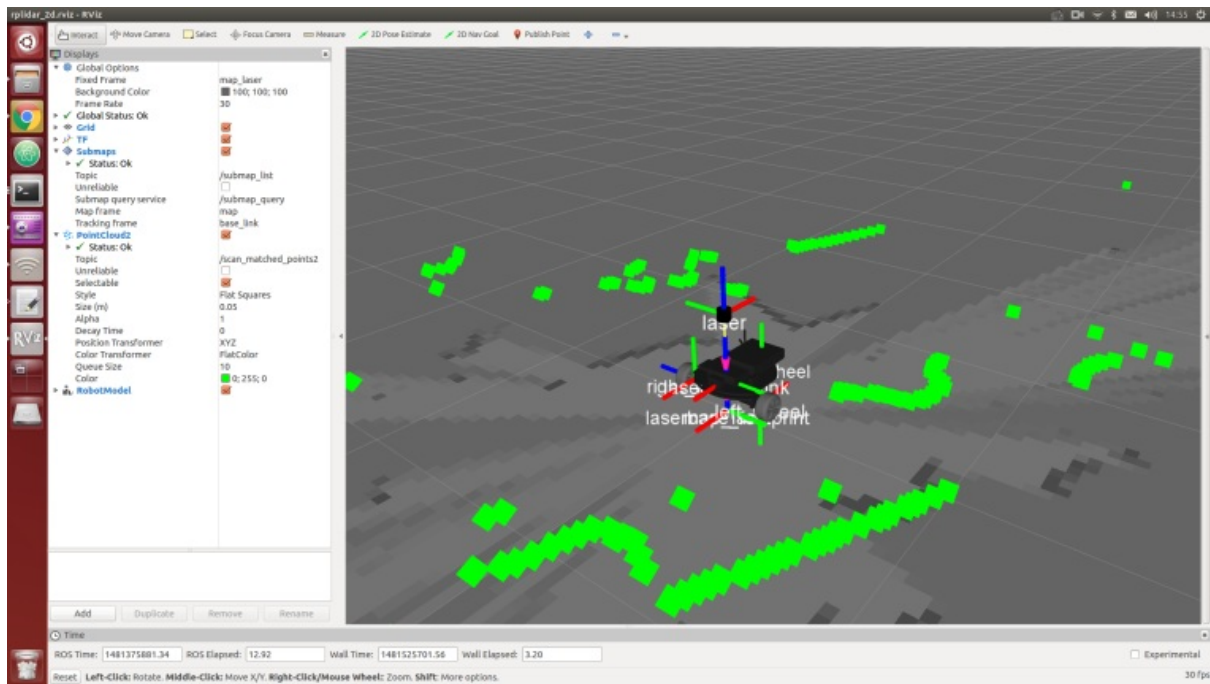
The newly recorded point bag file is in the home directory of Xiaoqiang, rename it to 1.bag

6. Start platformtographer_ros start bag playback build

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



7. If everything is normal, you can now see a similar effect in the picture below, waiting for the bag to be played.



8. Save the map, this article ends

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

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(17\) using ORB_SLAM2 to create a three-dimensional model of the environment](#)
 - [Using ORB_SLAM2 to create a three-dimensional model of the environment](#)
 - [Prepared work](#)
 - [Start ORB_SLAM2](#)
 - [Began to create an environmental 3D model](#)
 - [Use rviz to view the map result](#)
 - [Save the map](#)
 - [Loading of maps](#)
 - [Post processing of maps](#)
 - [ORB_SLAM2 parameter explanation](#)
 - [FAQ](#)

xiaoqiang tutorial (17) using ORB_SLAM2 to create a three-dimensional model of the environment

[Xiaoqiang Homepage](#)

Using ORB_SLAM2 to create a three-dimensional model of the environment

To achieve visual navigation, a three-dimensional model of space is required. [ORB_SLAM2](#) is a very effective algorithm for building spatial models. This algorithm is based on the recognition of ORB feature points, with high accuracy and high operating efficiency. We have modified the original algorithm, added the map's save and load functions, making it more applicable to the actual application scenario. The following describes the specific use.

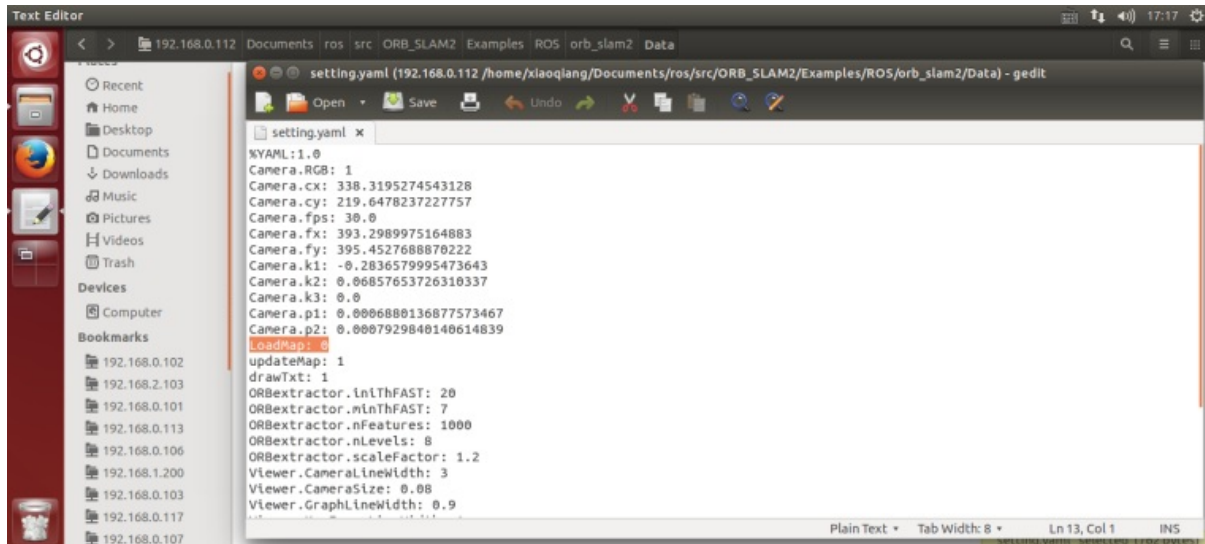
Note: Since the ORB_SLAM2 version of xiaoqiang has been upgraded to the Galileo version, the runtime detects if a valid certificate is available. Xiaoqiang is configured with a valid certificate. If you do not have a certificate by default, and you can contact customer service to get a cert for free. If you are not a xiaoqiang user, then the following tutorial cannot be performed.

Prepared work

Before starting ORB_SLAM2, please confirm that Xiaoqiang's camera is working properly. When building map with ORB_SLAM2, xiaoqiang has to move around, it is inconvenient to display the ORB_SLAM2 status at the same time (ssh remote operation is not smooth, and it is impossible to drag the monitor while running), please install Xiaoqiang map remote control [windows client](#) to get a better experience.

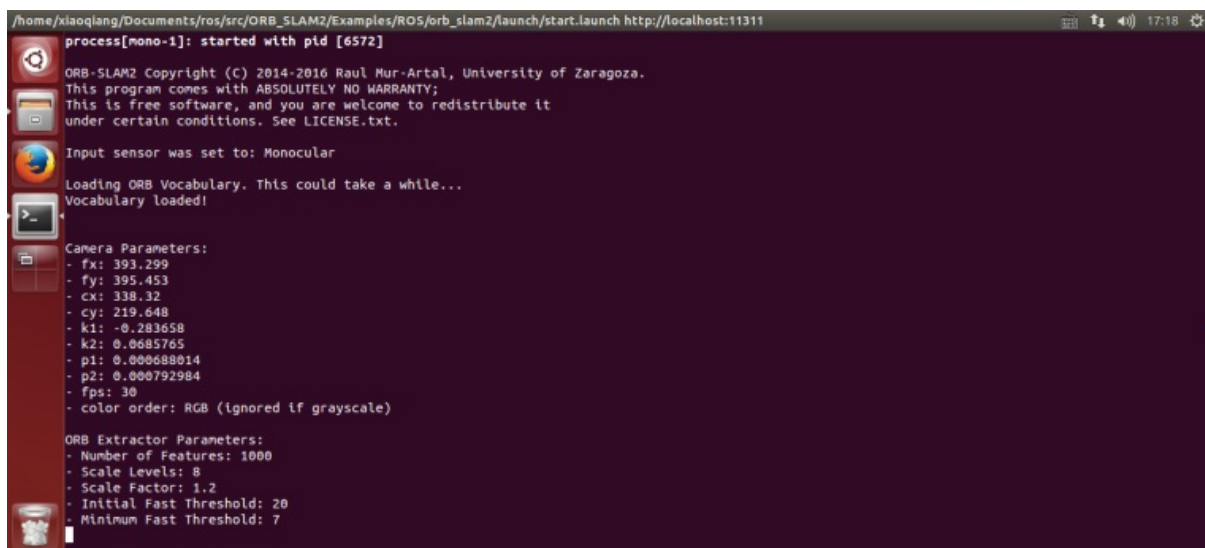
Start ORB_SLAM2

Change the configuration file. The configuration file for ORB_SLAM2 is located in the `/home/xiaoqiang/Documents/ros/workspace/src/ORB_SLAM2/Examples/ROS/ORB_SLAM2/Data` folder. Change the value of `LoadMap` in the `setting4.yaml1` and set it to 0. When set to 1, the program automatically loads map data from the `/home/xiaoqiang/slamdb` folder after startup. When set to 0, map data will not be loaded. Since we want to create a map, `LoadMap` needs to be set to 0.



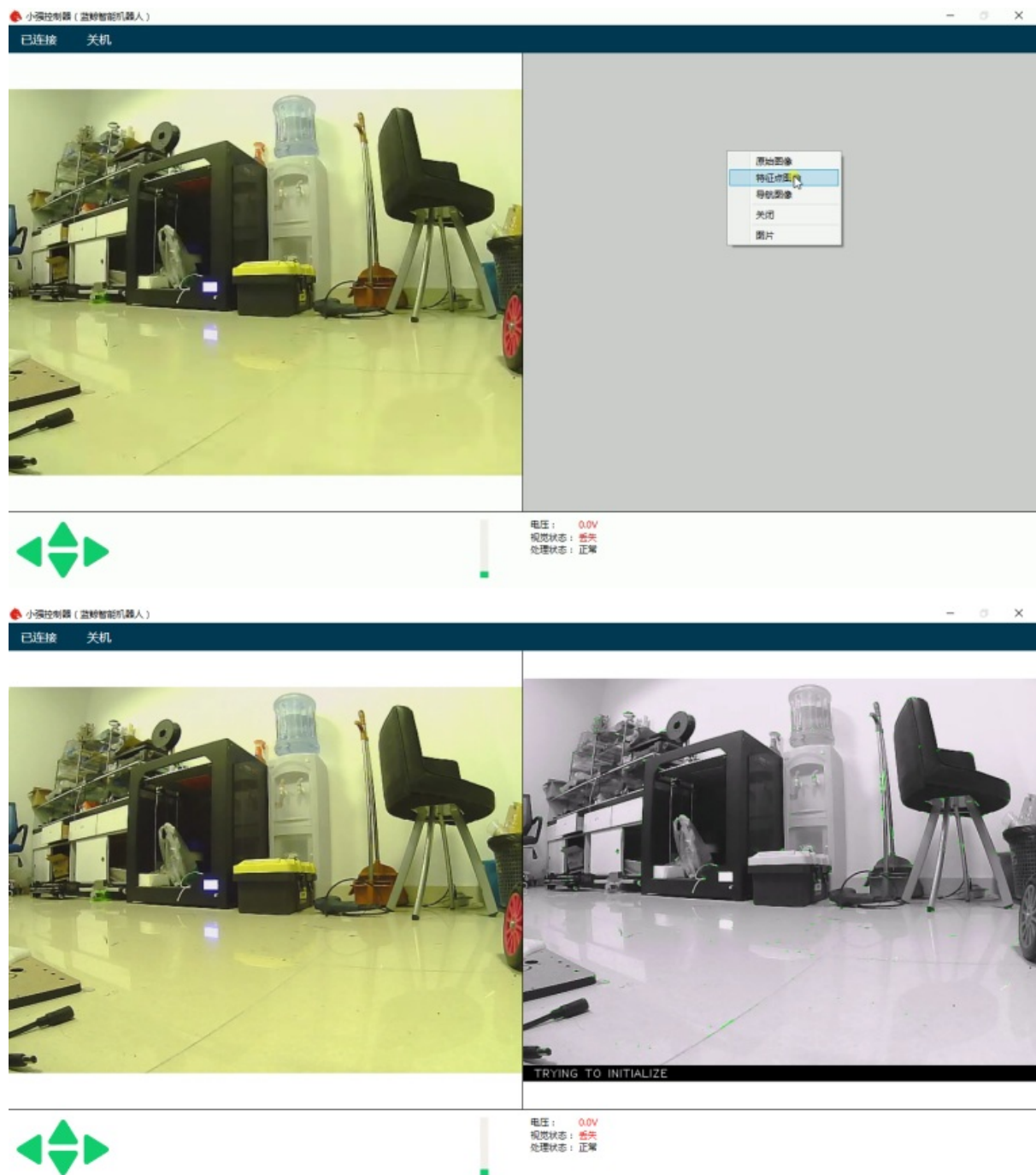
Use ssh to enter Xiaoqiang and execute the following command

```
ssh -X xiaoqiang@192.168.xxx.xxx
roslaunch ORB_SLAM2 map.launch
```

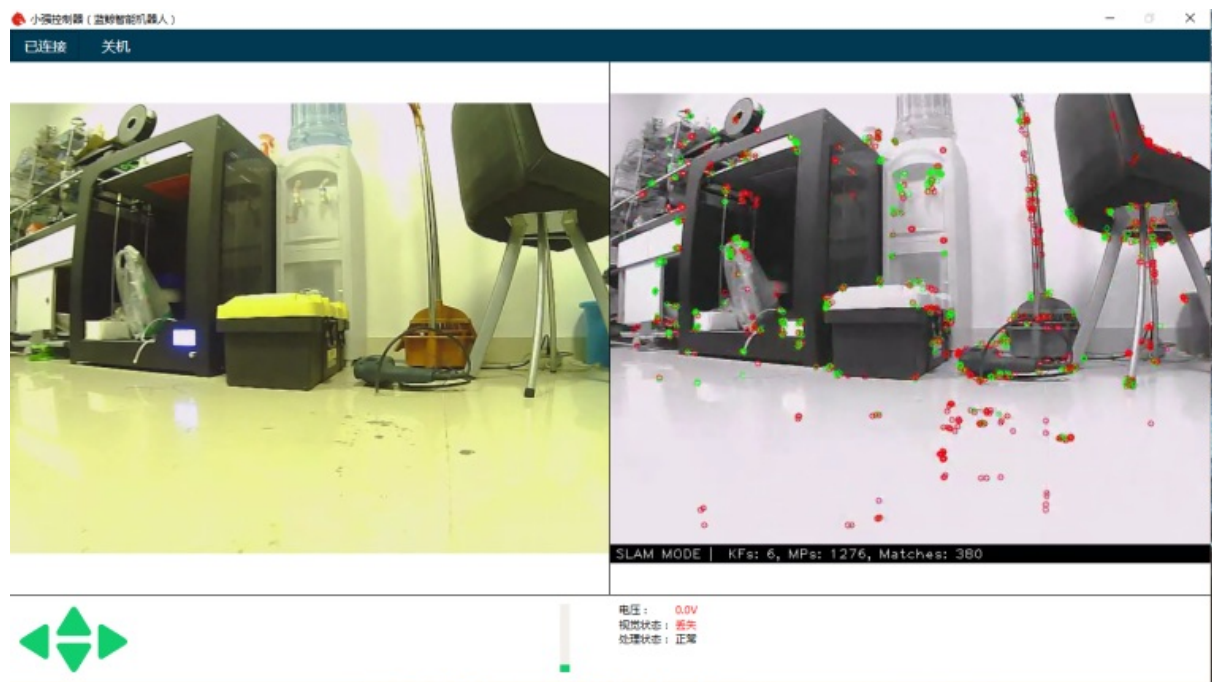


Began to create an environmental 3D model

Start Xiaoqiang remote control windows client, click on the "Not connected" button to connect Xiaoqiang. Right-click in the image window to open the "original image" and "ORB_SLAM2's feature point image"



In the figure above, the left image is the camera's original color image, and the right side is the ORB_SLAM2 processed black and white image. ORB_SLAM2 has not been initialized successfully in the image, so the black and white image has no feature points. Hold down the "w" key to start the remote control Xiaoqiang, slowly move forward, so that ORB_SLAM2 initialized successfully, that is, black and white images began to appear red and green features.



It is now possible to remote control of Xiaoqiang, and create a map of the surrounding environment. During the remote control process, it is necessary to ensure that the black-and-white image always has red and green feature points. If it does not exist, it means that the visual had lost, and it is necessary to remotely control Xiaoqiang to return to the place where the last missed lost position.

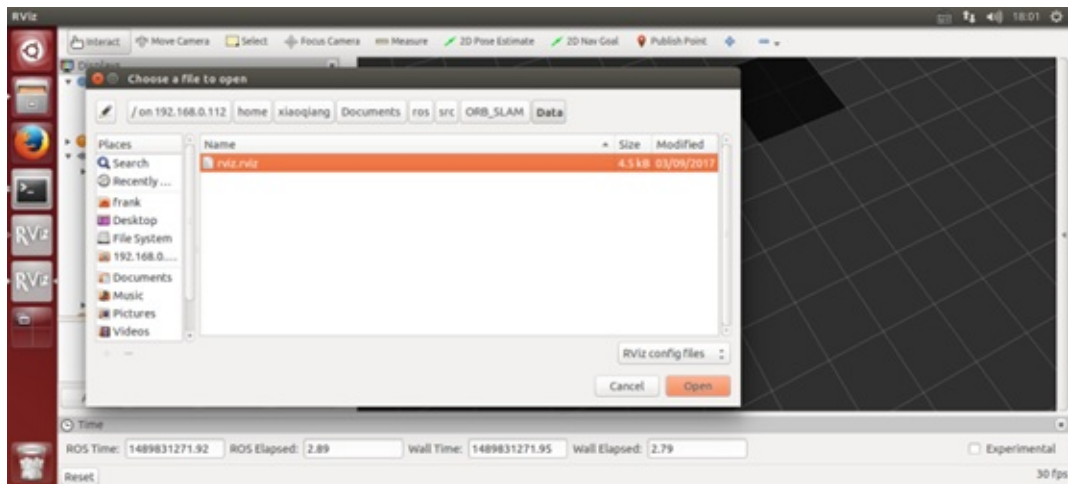


Use rviz to view the map result

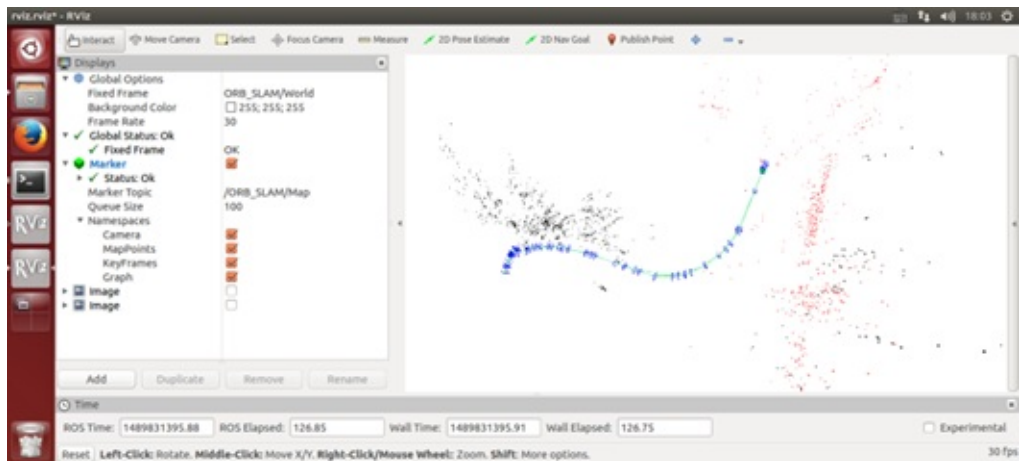
Add Xiaoqiang IP address in the local virtual machine's hosts file, then open a new terminal to open rviz.

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

Open the `/home/xiaoqiang/Documents/ros/src/ORB_SLAM/Data/rviz.rviz` configuration file. Some systems may not be able to open this file remotely. You can copy the file locally and open it locally.



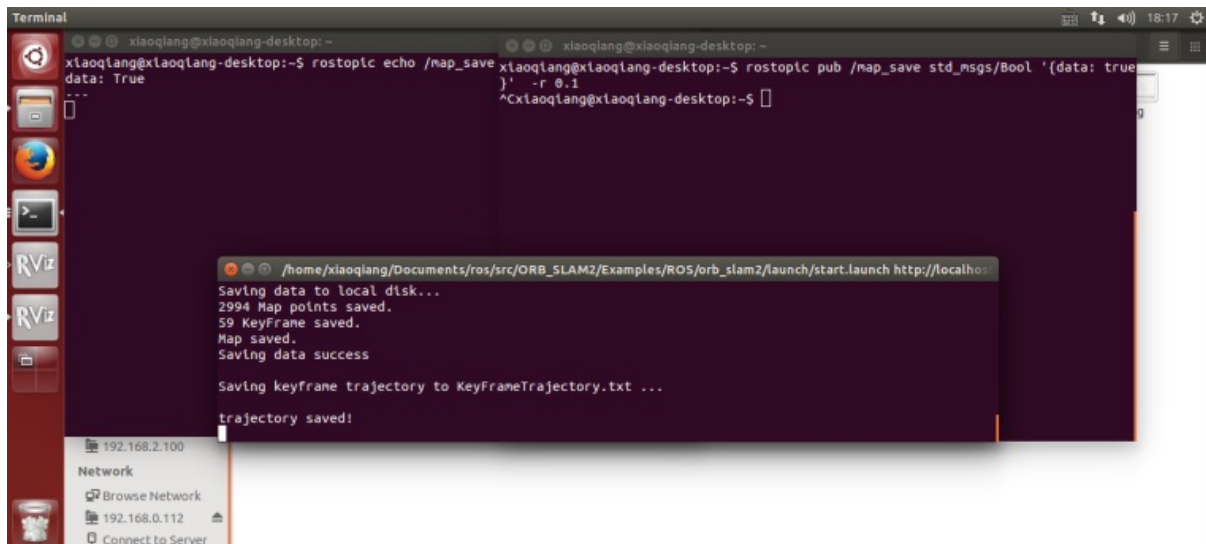
As shown in the figure below, the red and black points are three-dimensional models (sparse feature point clouds) created, and the blue boxes are keyframes that can represent the tracks of Xiaoqiang.



Save the map

After the map has been created to meet the requirements, start a new terminal in the VM and enter the following command to save the map:

```
ssh -X xiaoqiang@192.168.xxx.xxx
rostopic pub /map_save std_msgs/Bool '{data: true}' -1
```

The map file will be created in `~/home/slamdb`.

Loading of maps

You can load the map into ORB_SLAM2 again after saving the map. After the map is loaded, the program can quickly locate the location of the camera. The way to load the map is also very simple.

```
roslaunch ORB_SLAM2 run.launch
```

The `Loadmap` was set to 1 by the launch file actually.

Post processing of maps

After creating the map, you may want to use this map for navigation, you need to do further operations on the map file. For example, to create a navigation path, and so on. This section can be found in the article [Visual Navigation Path Editor Tutorial](#).

For xiaoqiang users, you can use Galileo navigation client, for more detail, please refer to [Galileo navigation user manual](#)

ORB_SLAM2 parameter explanation

The following is an ORB_SLAM2 setting file. The specific meaning of the parameter can refer to the comment.

```
%YAML:1.0
Camera.RGB: 1
# Camera calibration parameters, the parameters has moved to usb_cam after 2018.6
Camera.cx: 325.0466266836216
Camera.cy: 238.9974356449355
Camera.fps: 30.0
Camera.fx: 395.6779619478691
Camera.fy: 397.4934332366201
Camera.k1: -0.2805598785308578
Camera.k2: 0.06685114307459132
```

```

Camera.k3: 0.0
Camera.p1: -0.0009688764167839786
Camera.p2: -0.0002636873513136712
LoadMap: 0 # whether to load map, 1 load and 0 not load
updateMap: 1 # whether to update map while running, 1 update and 0 not update.
drawTxt: 1 # whether to display text in processed video
# ORB_SLAM2 slam algorithm-related parameters
ORBextractor.iniThFAST: 20
ORBextractor.minThFAST: 7
ORBextractor.nFeatures: 2000
ORBextractor.nLevels: 8
ORBextractor.scaleFactor: 1.2
# ORB_SLAM2 display-related parameters
Viewer.CameraLineWidth: 3
Viewer.CameraSize: 0.08
Viewer.GraphLineWidth: 0.9
Viewer.KeyFrameLineWidth: 1
Viewer.KeyFrameSize: 0.05
Viewer.PointSize: 1000
Viewer.ViewpointF: 500
Viewer.ViewpointX: 0
Viewer.ViewpointY: -0.7
Viewer.ViewpointZ: -1.8
UseImu: 1 # whether to use IMU, 1 use and 0 not use.
updateScale: 1 # Whether update scale while running, if set 1, the visual scale will be up
dated according to the data from odom
DealyCount: 5
# tf between camera and base_link
TbaMatrix: !!opencv-matrix
  rows: 4
  cols: 4
  dt: f
  data: [ 0., 0.03818382, 0.99927073, 0.06, -1., 0., 0., 0., 0., -0.99927073, 0.03818382,
0.0, 0., 0., 0., 1.]
# data: [ 0., 0.55176688, 0.83399839, 0., -1., 0., 0., 0., 0., -0.83399839, 0.55176688,
0., 0., 0., 0., 1.]
EnableGBA: 1 # whether to enable Global Bundle Adjustment
dropRate: 2 # frame skip rate, skip frames to improve performance
minInsertDistance: 0.3 # min distance between frames, prevent keyframes from being too den
se. Insert only one frame at this distance.
EnableGC: 0 # Whether to enable memory recycling, 0 is disable, 1 is enable. Enable memory
recycling can make more efficient use of memory, and the relative computational efficienc
y will decrease

```

FAQ

Q: Can I create map without the client?

A: Yes, you can. The client is mainly for remote control and video streaming. You can use command line remote control instead. But the client is recommended for better experience.

Q: Can I choose not to use IMU

A: The ORB_SLAM2 is an edited version with additional IMU support. If you don't want to use it, you can set UseImu to 0 in setting.yaml.

Q: Where is xiaoqiang's camera calibration file?

A:the camera calibration file located at `usb_cam/launch/ov2610.yaml`

Q: The camera's image is not clear

A:It may be that the camera's lens has been touched, causing it no focus. You can turn the camera lens and adjust the image to a clear position. Note that the camera parameters need to be re-calibrated after adjustment.

Calibration methods can be found in [how to calibrate camera](#)

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(18\) 3D modeling using DSO_SLAM](#)
 - [3D modeling using DSO_SLAM](#)
 - [1 . DSO installation](#)
 - [2. Installation of dso_ros](#)
 - [3.Start using](#)

xiaoqiang tutorial (18) 3D modeling using DSO_SLAM

[Xiaoqiang Homepage](#)

3D modeling using DSO_SLAM

[Direct Sparse Odometry \(DSO\)](#) was developed by Jakob Engel with better measured performance and accuracy than `lsd_slam`. DSO was open sourced to github by the author. At the same time, the author has also open sourced the usage code `dso_ros` of DSO in the ros system. This tutorial will demonstrate how to install DSO and `dso_ros` on the Xiaoqiang development platform, use the camera on Xiaoqiang platform to run DSO in real time to perform 3D modeling, Click to view the video.



1 . DSO installation

Note: Since the Xiaoqiang development platform has already installed a lot of DSO-needed dependencies in advance, the following will skip the installation of these packages. Readers of other development platforms should refer to the complete installation tutorial on github for installation.

1.a Installation Dependency Package

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

1.b download source code

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

1.c continue to configure dependencies

```
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 compile and install

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

2. Installation of dso_ros

Note: The source code provided by the original author has two branches. The master branch corresponds to the rosbuilt version, and the catkin branch corresponds to the catkin version. For modern ROS versions, the catkin version is recommended for easier installation and use. However, the author's catkin branch has a code defect and cannot be installed and used. Therefore, the following will install the dso_ros version of our Bwbot modification.

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

3. Start using

Note: The camera calibration files of the Xiaoqiang development platform are the same, so you can directly run the following commands. Readers of other development platforms should modify the contents of the camera.txt file (note that there should be no spaces at the end of each line) and Image topic name in commands.

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

Now move the camera and start modeling the surroundings in 3D, avoiding sharp turns and strenuous movements. For Xiaoqiang users, first control the Xiaoqiang movement and use the rosbag to record the image/camera_node/image_raw image topic data, and then replay. This can achieve a wide range of

modeling. Before the rosbag is replayed, it is necessary to stop the camera node. `sudo service startup stop` , otherwise there will be image publishing conflicts.

[Xiaoqiang Homepage](#) [Back To Index](#)

- [Xiaoqiang tutorial \(19\) usage of Nllinepatrol_planner](#)
 - [1. Configure Nllinepatrol_planner](#)
 - [2. Make a move_base launch file](#)
 - [3. Start using after configuration](#)
 - [4. Now that the target global path trajectory \(green line\) has appeared in rviz and you want to test other goal targets, modify the code in Nllinepatrol.py.](#)

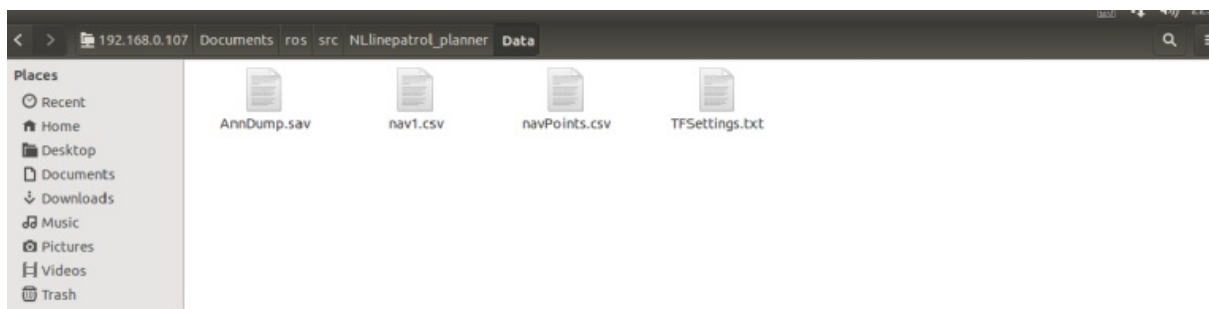
Xiaoqiang tutorial (19) usage of Nllinepatrol_planner

[Xiaoqiang Homepage](#)

The `Nllinepatrol_planner` included with Xiaoqiang's host is a global path planner for visual navigation. According to Xiaoqiang's output visual trajectory (for visual trajectory files, please refer to this [post](#)), it can output a global path link to Xiaoqiang's current position and destination target point. It will be demonstrated below using a simulation example. The main idea is: a python script publishes virtual visual odometers and related tf trees. Another python script publishes target points to the `move_base` node. Finally, the `move_base` node obtains a global path by calling `Nllinepatrol_planner` and displays it in rviz.

1. Configure Nllinepatrol_planner

To use `Nllinepatrol_planner`, you need to provide visual trajectory file that `Nllinepatrol_planner` will read and (transformation parameters) file that is required for the coordinate transformation of the track, both files should be placed in the data folder under `Nllinepatrol_planner`, the file name is arbitrary, by configuring the related parameters in `move_base`. You can specify the file that `Nllinepatrol_planner` reads, as explained below.



In the figure above, `nav1.csv` is the visual trajectory file and `TFSettings.txt` is the transformation parameter file (the first row is the 9 elements of the rotation matrix, the array elements are arranged in the row of the c language, and the second row is the xyz of the translation vector. Components, the third one is the scale factor)

2. Make a move_base launch file

In this tutorial, we have provided the relevant launch file in the launch folder of the `nav_test` package. The file name is `xq_move_base_blank_map2.launch`. This launch file can be used as a template during actual use. Please pay attention to the following figure.

```

xq_move_base_blank_map2.launch x
<launch>
  <node pkg="tf" type="static_transform_publisher" name="baselink_broadcaster" args="0 0 0.13 0 0 1 base_footprint base_link 10"/>
  <renop from="/odom" to="/ORB_SLAM/Odom" />
  <!-- Run the map server with a blank map -->
  <node name="map_server" pkg="map_server" type="map_server" args="$(find nav_test)/maps/blank_map.yaml"/>
  <include file="$(find nav_test)/launch/xq_move_base2.launch" />
  <!-- Run a static transform between /odom and /map -->
  <node pkg="tf" type="static_transform_publisher" name="odom_map_broadcaster" args="0 0 0 0 0 /map /odom_combined 100" />
</launch>

```

视觉里程计topic, 实际使用时由orb_init中的orb_scale.py文件产生
实际使用时删除, 已经在小强串口驱动里发布了
这个transform很关键

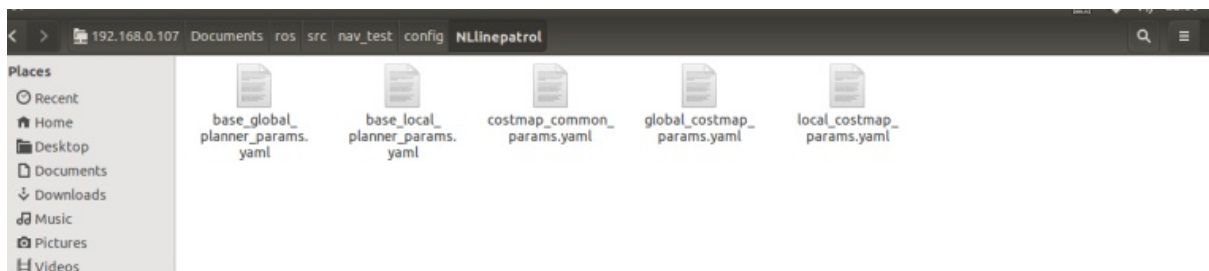
The launch file will call the `xq_move_base2.launch` file. The `xq_move_base2.launch` file is also in the current directory. The contents are as follows:

```

<launch>
  <node pkg="move_base" type="move_base" respawn="false" name="move_base" output="screen"
  >
  <param name="base_global_planner" value="Nllinepatrol_planner/NllinepatrolPlanner"/>
  <rosparam file="$(find nav_test)/config/Nllinepatrol/costmap_common_params.yaml" comma
  nd="load" ns="global_costmap" />
  <rosparam file="$(find nav_test)/config/Nllinepatrol/costmap_common_params.yaml" comma
  nd="load" ns="local_costmap" />
  <rosparam file="$(find nav_test)/config/Nllinepatrol/local_costmap_params.yaml" command
  ="load" />
  <rosparam file="$(find nav_test)/config/Nllinepatrol/global_costmap_params.yaml" comma
  nd="load" />
  <rosparam file="$(find nav_test)/config/Nllinepatrol/base_local_planner_params.yaml" c
  ommand="load" />
  <rosparam file="$(find nav_test)/config/Nllinepatrol/base_global_planner_params.yaml"
  command="load" />
</node>
</launch>

```

Through the above, it is found that by setting the value of the `base_global_planner` parameter to specify the global path planner as `Nllinepatrol_planner`, it can also be seen that the other parameter configuration files of `move_base` are stored in the `config/Nllinepatrol` path in the `nav_test` package.



For the above image, the file we need to change is `base_global_planner_params.yaml`, because the contents of this file correspond to the parameters actually loaded by the `Nllinepatrol_planner` runtime. The default content is as follows:

```

NllinepatrolPlanner:
  DumpFileName: AnnDump.sav
  strTFParamsFile: TFSettings.txt
  TxtFileName: nav1.csv
  ANN_Dump_Bool: false

```



```
connect_distance: 0.3
```

TxtFileName specifies the name of the loaded visual track file, strTFParamsFile specifies the name of the loaded transformation parameter file, and connect_distance sets the maximum distance between the connected points in the visual track file. (The distance between two points after the coordinate transformation is smaller than the value is considered as There is no obstacle between the two points, you can directly connect),

ANN_Dump_Bool value is false to load the trajectory and transformation parameters from the txt file, if it is true then load from the dump file specified by the DumpFileName parameter (When using the same visual track file multiple times , start from the dump file after the second time can be accelerated)

3. Start using after configuration

A. Because we are virtual running this time, some of the published topics are of no practical significance but conflict with Xiaoqiang's default ROS driver. So now we need to stop all ROS running instances.

```
sudo service startup stop
roscore
```

B. Start virtual topic and Xiaoqiang model files

```
roslaunch orb_init temp.py // Publish odom
roslaunch xiaoqiang_udrf xiaoqiang_udrf.launch // Start the model
```

C. Launch the xq_move_base_blank_map2.launch file produced above

```
roslaunch nav_test xq_move_base_blank_map2.launch
```

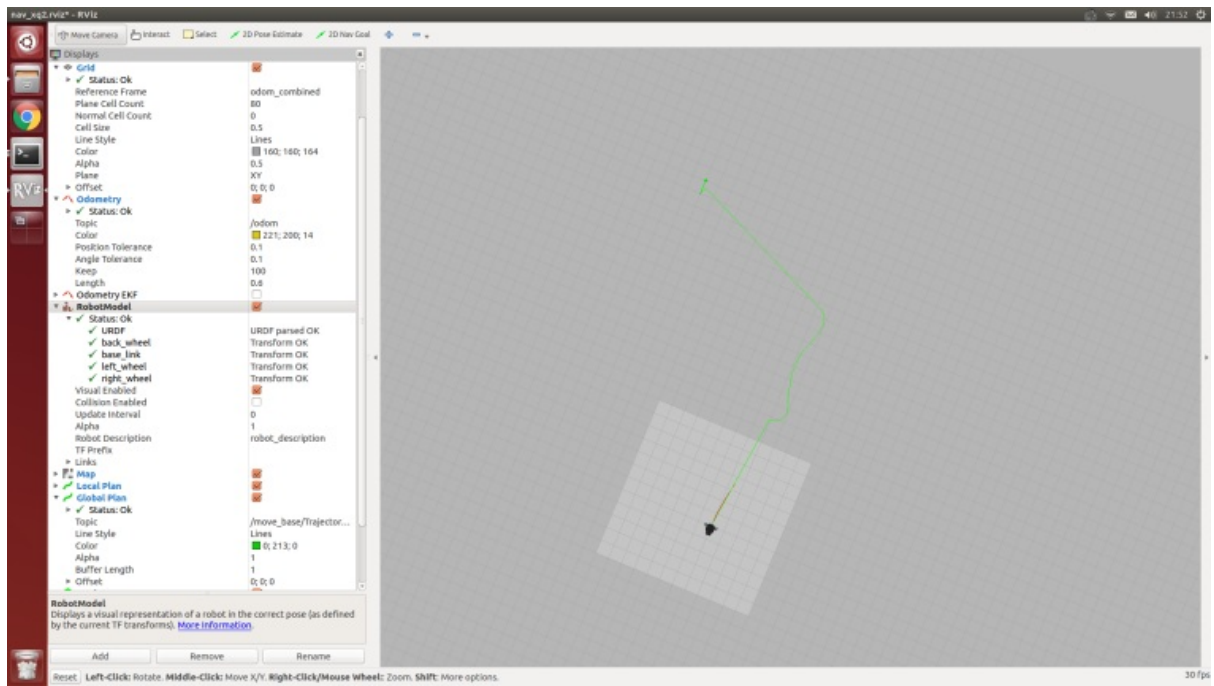
D. Start rviz and open the ros/src/nav_test/config/nav_xq2.rviz configuration file

```
rviz
```

E. Launch virtual goal publishing node (based on squire.py modification in inertial navigation)

```
roslaunch nav_test NLlinepatrol.py
```

4. Now that the target global path trajectory (green line) has appeared in rviz and you want to test other goal targets, modify the code in NLlinepatrol.py.



Xiaoqiang Homepage [Back To Index](#)

- [xiaoqiang tutorial \(20\) get vision odometer and display the xiaoqiang track in the rviz](#)

xiaoqiang tutorial (20) get vision odometer and display the xiaoqiang track in the rviz

[Xiaoqiang Homepage](#)

1. Local ssh preparation section (Xiaoqiang host is the controlled end, local refers to remote control end)

ssh remote login Xiaoqiang host, the following operation is entered in this ssh window if without special statement

```
ssh xiaoqiang@xxx.xxx.xxx.xxx
# Please replace xxx.xxx.xxx.xxx with Xiaoqiang's current actual ip address
# Start multi-window manager
screen
```

Screen usage please refer to this [tutorial](#)

Check if the `orb_init` package has installed

```
rospack find orb_init
```

Delete if it exists

```
cd ~/Documents/ros/src
rm -r orb_init
```

Download the latest version of `orb_init` from the Bluewhale Open Source Software Store and install it

```
cd ~/Documents/ros/src
git clone https://github.com/BluewhaleRobot/orb_init.git
```

Check whether startup task is running

```
sudo service startup status
```

If running is displayed, it means normal. If stopped is displayed, then restart it

```
sudo service startup start
# If you want to close this task, you can use this command
sudo service startup stop
```

Check system status

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

If it is normal, the display is as follows

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

If it is abnormal, restart the startup task.

```
sudo service startup restart
```

Start ORB_SLAM in another command window in screen

```
roslaunch ORB_SLAM ov2610.launch
```

Return to the previous screen window and wait for ORB_SLAM to start

```
rostopic echo /system_monitor/report
# If ORB_SLAM is started, the following is displayed
orbStartStatus: True
```

2. Native local operation section

This machine has been installed ros jade version of the robot system, the computer operating system is ubuntu14.04, ros can be installed to refer to this [tutorial](#).

Add this machine to Xiaoqiang's ros network, open a command line terminal locally, and add Xiaoqiang's ip in the local hosts file.

```
sudo gedit /etc/hosts
Add to
xxx.xxx.xxx.xxx xiaoqiang-desktop
Save and exit
Please replace xxx.xxx.xxx.xxx with Xiaoqiang's current actual ip address
```

Join ros LAN

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

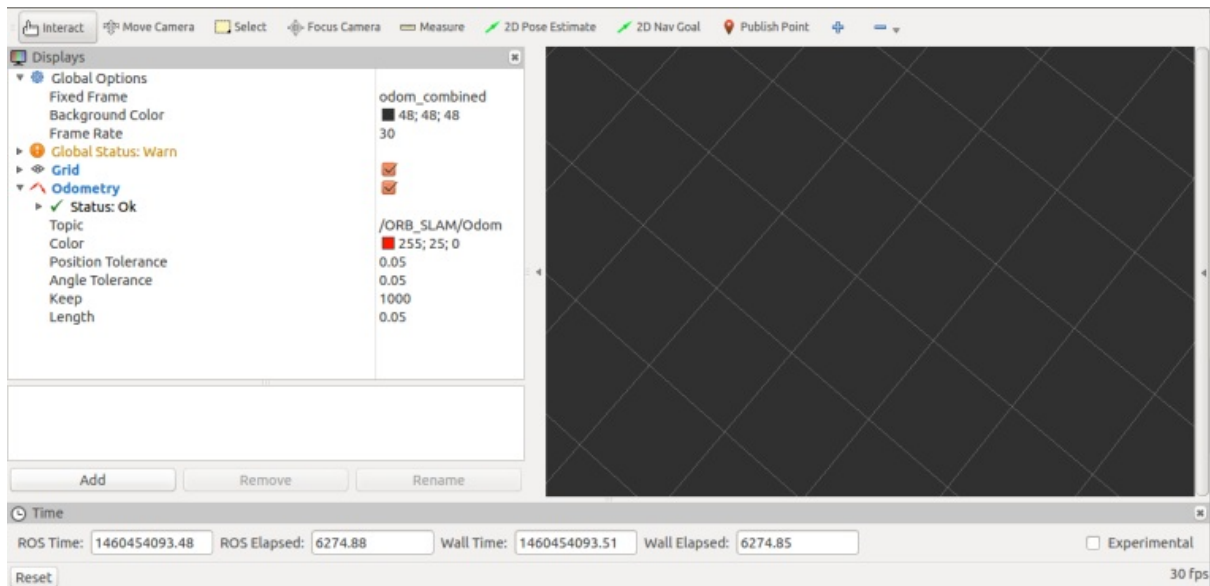
If the join is successful, the command line will output the topic on Xiaoqiang's host. For more information about setting up multiple ros machines online, please refer [here](#).

Download the rviz [configuration file](#). This configuration file can also be copied directly from the orb_init package on Xiaoqiang's host to view the output path of the Xiaoqiang vision system.

Enter in local command line terminal

```
rviz
```

When the window opens, click on file->open in the upper left corner and select the above downloaded configuration file. At this time the interface should appear as shown below



3. In ssh in the screen to open a new window, used to start orb_init Xiaoqiang host, before the start to ensure Xiaoqiang around two square meters of free space, Xiaoqiang will move for some time

```
roslaunch orb_init orb_scale.py
```

After orb_init initialization is completed, orb_init cannot be closed. It will continue to output the visual odometer topic. This topic is the content that the local rviz needs to display. At this time, a new window is opened to check the system status.

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

If it is normal, the display is as follows

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

At this point we have already acquired Xiaoqiang's visual odometer

4. In ssh in the screen to open a new window, used to control Xiaoqiang move

```
roslaunch nav_test control.py
```

Use the arrow keys to control the movement of Xiaoqiang. Spacebar is stopped. Ctrl + C exits the program.

5. As Xiaoqiang moves, the rviz interface on the local machine will be updated to show Xiaoqiang's trajectory in real time. Our own test video is [here](#). For more information on how to use rviz, see [here](#).

[Xiaoqiang Homepage](#) [Back To Index](#)

- [xiaoqiang tutorial \(21\) get USB camera 30fps 1080p image stream and 120fps VGA resolution image stream](#)
 - [1.Upgrade usb_cam node source code](#)
 - [2.Test 1080p video output at 30fps](#)

xiaoqiang tutorial (21) get USB camera 30fps 1080p image stream and 120fps VGA resolution image stream

[Xiaoqiang Homepage](#)

Xiaoqiang's default USB camera node publishes a 30fps VGA (640*480) image stream that can satisfy most visual tasks. The USB camera hardware can actually output 120fps VGA image stream and up to 30fps 1080p image stream. This article will introduce how to obtain these two type of the image.

1.Upgrade usb_cam node source code

Taking into account the USB2.0 bandwidth, output 120fps VGA video and 30fps 1080p video, need to use mjpeg format. For Xiaoqiang's own USB camera ROS driver package `usb_cam`, it can not read Xiaoqiang usb camera in mjpeg way, so we need to upgrade `usb_cam` first. For the upgrade method, please refer to [this post](#)

2.Test 120fps VGA video output

First please turn off Xiaoqiang's startup task

```
sudo service startup stop
```

Modify the contents of the `ov2610mjpg.launch` file in the `usb_cam` package as shown below

```
<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>
```

Launch the above launch file

```
roslaunch usb_cam ov2610mjpg.launch
```

Open a new window to print the published image topic frame rate

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

Normal output similar to the following figure

```
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
```

The above output shows that the camera frame rate is about 99fps and does not reach 120fps. This is due to the influence of ambient light, which can reach 120 frames if the ambient light is sufficient.

2. Test 1080p video output at 30fps

First please stop Xiaoqiang's startup service

```
sudo service startup stop
```

Modify the contents of the `ov2610mjpg.launch` file in the `usb_cam` package as shown below.

```
<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>
```

Launch the above launch file

```
roslaunch usb_cam ov2610mjpg.launch
```

Open a new window to print the published image topic frame rate.

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


Normal output similar to the following figure.

```
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
average rate: 29.849
  min: 0.027s max: 0.042s std dev: 0.00282s window: 202
```

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(22\) Operating 6 DOF robotic arm](#)
 - [1. Run the robot_arm node](#)
 - [2. Construct robot_arm/cmdstring topic](#)
 - [3. Publish motion commands](#)
 - [4. There is no video for the sport result, you need to operate it yourself. If you feel unsatisfactory with the movement, please refer to our servo control posts: \[电机控制与缓动函数\]\(#\)](#)

xiaoqiang tutorial (22) Operating 6 DOF robotic arm

[Xiaoqiang Homepage](#)

6 degrees of freedom robot arm [resources](#). Please read the serial communication protocol of the secondary development part in the resources. Although the following is the usb hid communication method, the command format is the same.

The following will operate the 6-DOF arm to complete the 3 sets of actions stored on the main control board.

Control principle: Xiaoqiang host uses the usb to connect the robot master control board, user sends a topic named `robot_arm/cmdstring`, this topic content is the control command, and the robot_arm node is responsible for sending the topic content to the master control board of the robot arm via USB protocol.

1. Run the robot_arm node

```
roslaunch robot_arm move.py
```

If the result is normal, the robot arm will perform action 0 (the default state), and the command window will show the following

```
xiaoqiang@xiaoqiang-desktop:~/Documents/ros$ roslaunch robot_arm move.py
Opening robot arm device
Manufacturer: MyUSB_HID
Product: LOBOT
Serial No: 8D9823654852
Run the zero group action
```

2. Construct robot_arm/cmdstring topic

`robot_arm/cmdstring` This topic type is `std_msgs/String`. According to the secondary development data of the robot arm provided above, we can know that the 6-DOF control protocol is represented by an unsigned byte array. Therefore, we slightly modify it here to directly convert the hex value of this array into a string.

Packaged into the topic command. The transformation method uses the byte array representation of python to string together the hex encodings of each element in the array, and then replace `0x` with `\x`

```
# E.g: [0x55,0x55,0x05,0x06,0x00,0x01,0x00] This control command array, converted to robot
_arm/cmdstring content is '\x55\x55\x05\x06\x00\x01\x00'
# Tip: If you don't understand it, you can use Python's map function to help convert it.
map(ord, '\x55\x55\x05\x06\x00\x01\x00')
```

We want to control the robot arm to complete 3 actions. The contents of these three strings are as follows:

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

3. Publish motion commands

Open a new terminal, because it is a demonstration, so we directly using the pub function of rostopic, the above string command packaged into a topic sent to the robot_arm node

```
# Action 1:
rostopic pub robot_arm/cmdstring std_msgs/String '\x55\x55\x05\x06\x00\x01\x00'
# Action 2:
rostopic pub robot_arm/cmdstring std_msgs/String '\x55\x55\x05\x06\x01\x01\x00'
# Action 3:
rostopic pub robot_arm/cmdstring std_msgs/String '\x55\x55\x05\x06\x02\x01\x00'
```

4. There is no video for the sport result, you need to operate it yourself. If you feel unsatisfactory with the movement, please refer to our servo control posts: [电机控制与缓动函数](#)

[Xiaoqiang Homepage Back To Index](#)

- [xiaoqiang tutorial \(23\) ROS introductions](#)

xiaoqiang tutorial (23) ROS introductions

[Xiaoqiang Homepage](#)

[Learning ROS for Robotics Programming - Second Edition.pdf](#)

This tutorial is very basic, although the book uses hydro as an example, but also fully compatible with the kinetic version. Just replace the hydro string in the code example in the book with kinetic. After completing the Xiaoqiang [tutorial \(1\)](#), if you are not familiar with ROS, please read Chapters 2 and 3 of this book.

[Xiaoqiang Homepage Back To Index](#)

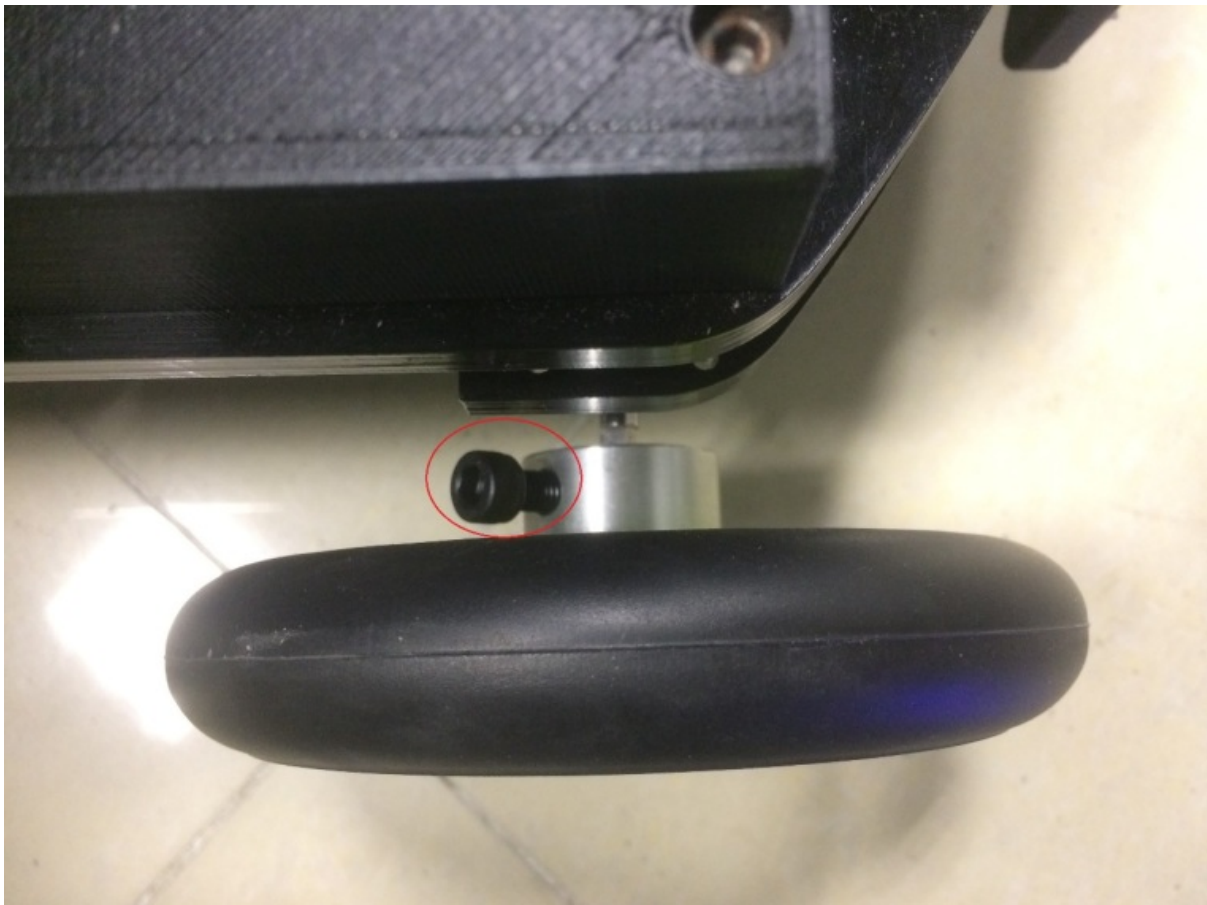
- [Daily maintenance instructions](#)
 - [Charging instructions](#)
 - [Wheel loosening solution](#)

Daily maintenance instructions

Charging instructions

After disconnecting the battery from the chassis, charge it with the dedicated battery charger. It takes about 5 hours to fully charge the battery, and the indicator will change from red to blue when full. The two output wires of the battery are connected in parallel, so both heads can be used for charging and discharging (the charger has only one male), and the battery supports simultaneous charge and discharge

Wheel loosening solution



Solution: Re-tighten the screws in the figure above

- [Xiaoqiang Bottom Driver Firmware Download and Upgrade Methods](#)

Xiaoqiang Bottom Driver Firmware Download and Upgrade Methods

Users received Xiaoqiang after March 11, 2017. Please refer to this post to upgrade the firmware. If you cannot upgrade, please ask for customer service before using the following method. Manually upgrade risk, please consult customer service before operation.

Xiaoqiang v4_3.hex ,Right click "Save As..." and download it as xiaoqiang.hex

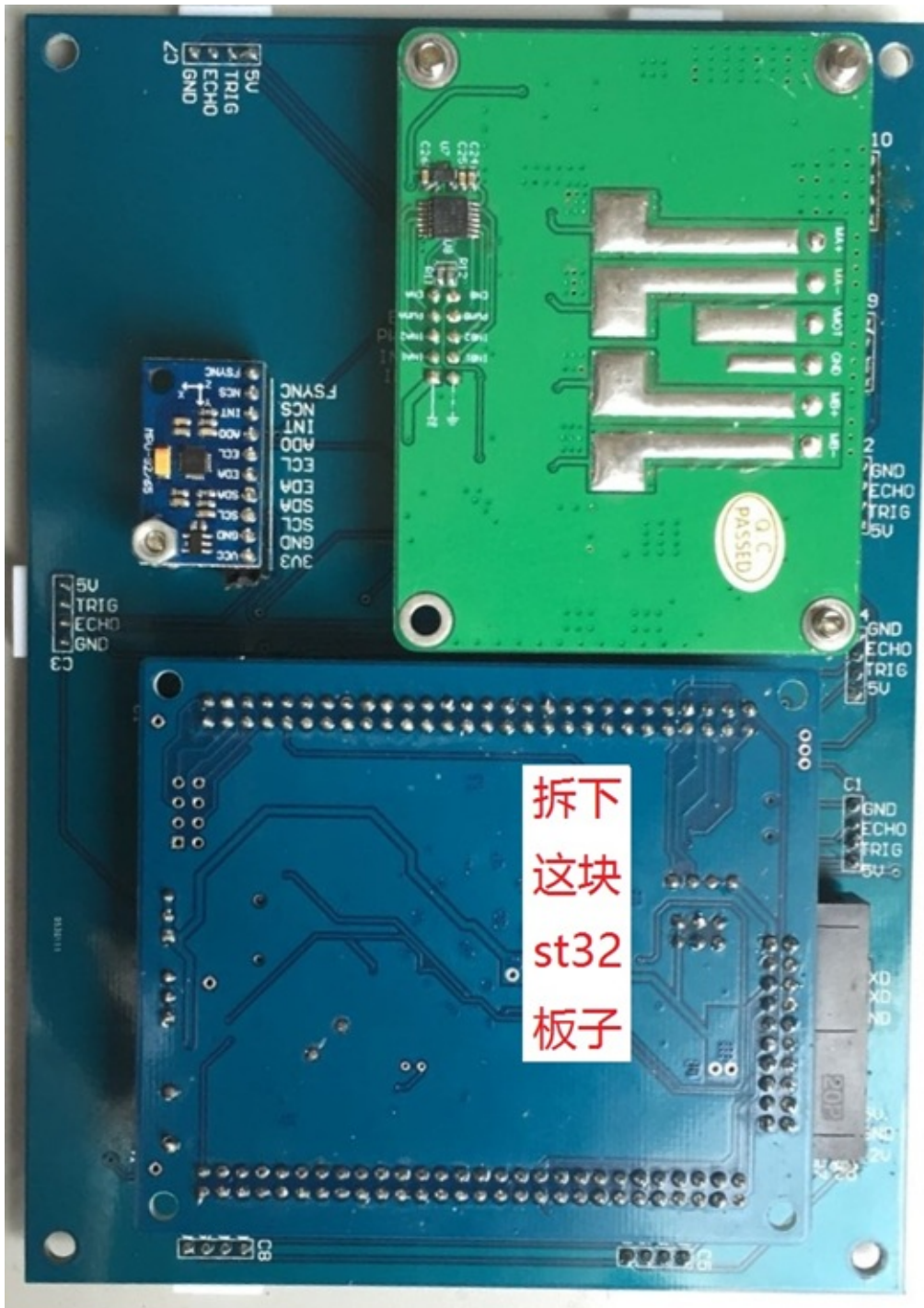
The firmware is an ordinary hex file and supports any stm32 downloader. After stm32 is burned, please complete the host computer update operation according to the upgrade chassis ros driver package xqserial_server. Finally recalibrate the chassis IMU according to this tutorial.

The following describes the use of Xiaoqiang's own method of using USB to serial port to write stm32. After this upgrade, you can use the simple upgrade method in this post.

1. Find a Windows computer, download and install this stm32flash software

flash_loader_demo_v2.8.0.exe , right click "Save as..." to start download

1. After power off the chassis, remove the VMOT GND wire on the middle of the green motor driver board, slowly pull out the stm32 core board on the chassis, and then adjust the two short caps on the board according to the following figure.

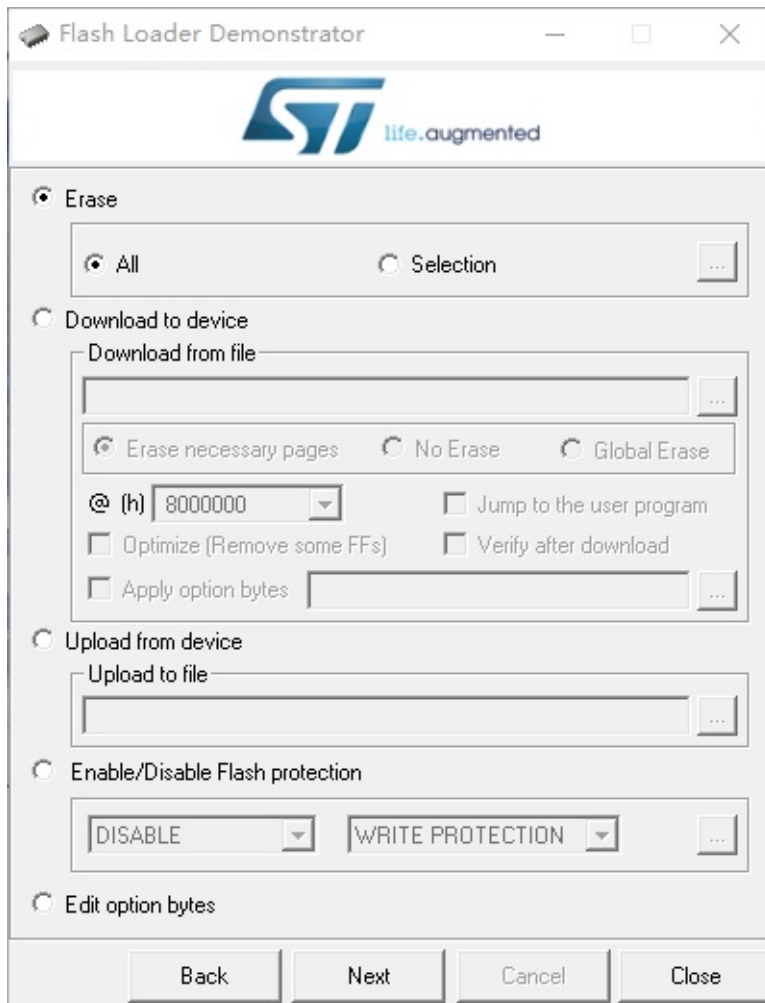




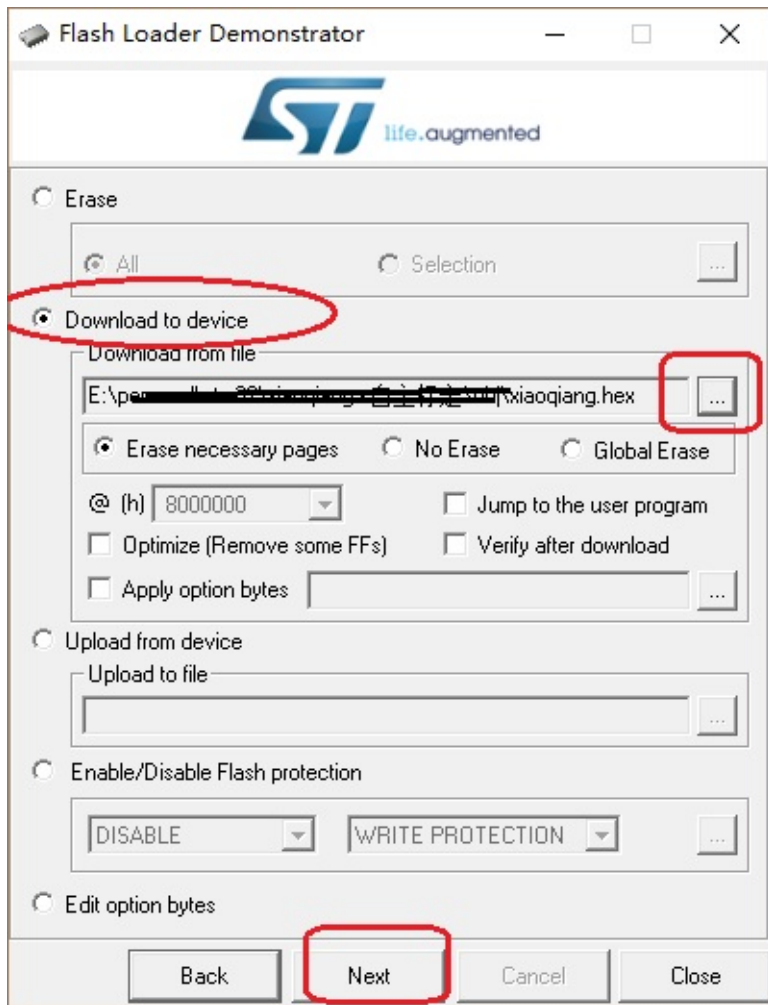
1. Insert the stm32 board after adjustment in the second step into the chassis. Note that the pins need to be fully aligned with the chassis, and then the chassis is powered on.
2. Unplug the usb head of the USB to serial port module on Xiaoqiang from the host computer and insert it into the windows computer. Open the Flash Loader Demo program from the Start Programs menu. All Programs->STMicroelectronics->Demonstrator GUI, the following interface appears.



1. Select the default settings, always point next (If you click next failure, re-power the chassis and try again), the following interface appears :

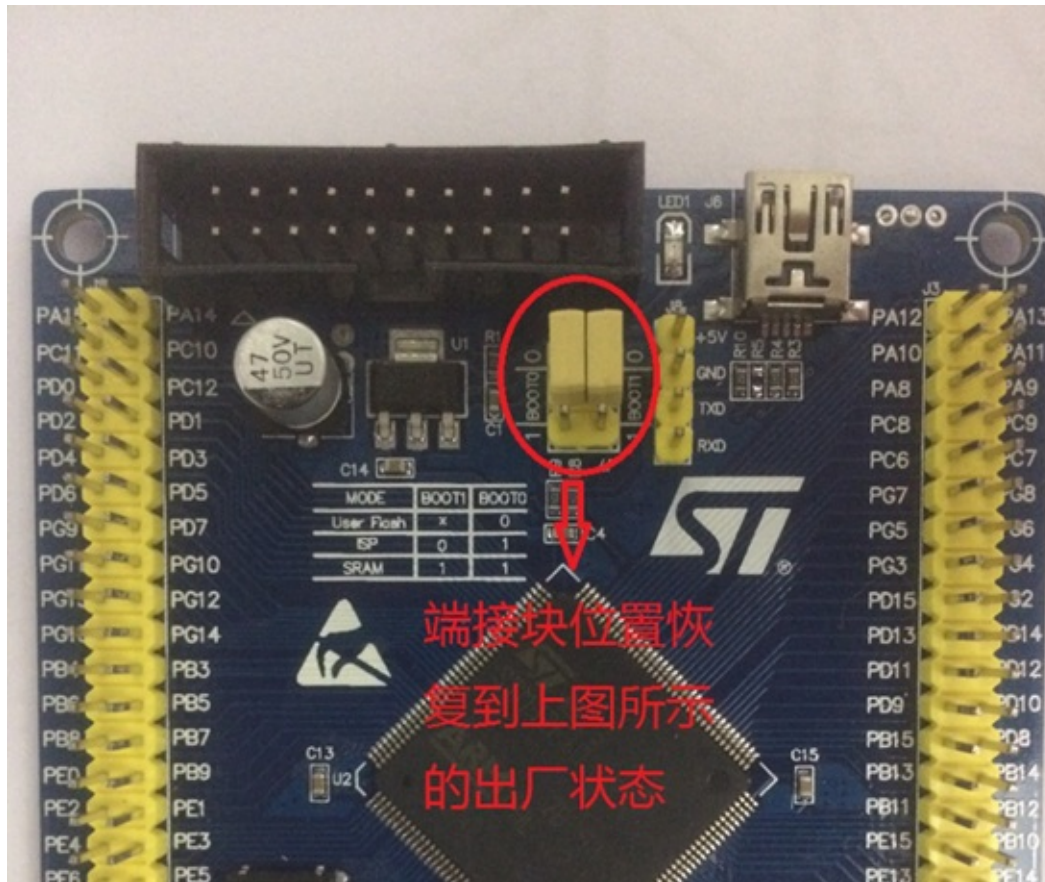


1. Select Download to device, select the hex file provided above, click Next to start the download.





1. After the burning is completed, please close the software, power off the chassis, and insert USB to serial port back to Xiaoqiang's host computer.
2. Slowly unplug the stm32 core board on the chassis again, adjust the two shorting caps on the board back to the initial state, insert it back into the chassis (pins need to be fully aligned with the insertion base), the green motor driver board The middle "VMOT GND" wiring is plugged in, stm32 upgrade is completed!



1. Chassis re-powered, according to the tips at the beginning of the article to complete the upgrade of the host computer.

- [Upgrade chassis ros driver package xqserial_server](#)

Upgrade chassis ros driver package xqserial_server

1. ssh login Xiaoqiang host, enter Xiaoqiang ros working directory

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

1. Enter ros driver package xqserial_server, update software

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

1. Restart ros node, update completed

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

- [Recalibrate the chassis IMU](#)

Recalibrate the chassis IMU

Sorry: This tutorial is for users who purchased after September 2016

Applicable situation:

Before shipment, each chassis IMU has been calibrated, and theoretically normal use does not require recalibration. If after a long period of use of the chassis, it is found that the odom angle at the output of the chassis begins to have a serious flow, follow the steps below to recalibrate the chassis IMU.

Steps:

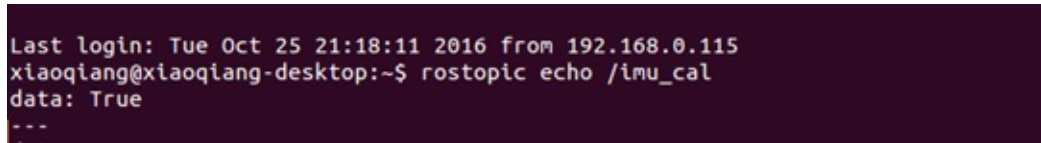
1. Place the chassis horizontally stationary, the following calibration process can not move or hit the chassis.
2. In the local virtual machine ssh login, enter the following command.

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

1. Open a new window in the local virtual machine and login to the host again. Enter the following command:

```
ssh xiaoqiang@192.168.0.xxx -X
rostopic pub /imu_cal std_msgs/Bool '{data: true}' -1
```

1. Wait 10 seconds. When the window in step 1 shows the following figure, it indicates that the calibration procedure has been started. Please terminate the topic issue command in step 2.



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

1. Wait 2 minutes, IMU re-calibration is completed, now you can continue normal use without restart.

- [Bluewhale ROS system image released](#)
 - [System configuration requirements](#)
 - [Latest Kinetic Image](#)
 - [Latest Jade Image](#)
 - [Image installation method](#)

Bluewhale ROS system image released

This article lists all published Bluewhale ROS system images, users can choose according to their needs. The Bluewhale ROS system image is the image of the ROS system created by Bluewhale Robot based on Ubuntu ROS. It contains the ROS system and some common ROS packages. Ideal for ROS learning and developers. After the installation of the system is complete, you do not need to install and configure ROS. You can use it directly. The Bluewhale ROS image is also the system image of Xiaoqiang and can be installed directly on Xiaoqiang.

Note: It's recommended to use [Vmware Player](#)

System configuration requirements

Minimum memory 1.5G recommended 2G, storage space at least 30G.

Latest Kinetic Image

`xq_os_v2.0.8-2018-09-11.iso` (kinetic version) can be used on Xiaoqiang Pro and your own computer.

[Download Address](#)

md5: 0a2d98bbf93d9556163df26fe42a072c

`xq_os_lungu_v2.0.5_2018-7-8` (kinetic version) can be used on Xiaoqiang XQ5 and can your own computer.

[Download address](#)

md5: 6e4dda157e2b96bf271f5f4e33e181e2

`xq_os_v2.0.6_mini_2018-07-19` (kinetic version) can be used on Xiaoqiang mini and your own computer.

[Download Address](#)

md5: 8696d1e74a10123568db0dab2c6e1e01

Latest Jade Image

`xq_os_v1.0.3_2017-10-31` Can be used on Xiaoqiang Pro and your own computer

[Download Address](#)

md5: afbf1c8026733c4f8063778846c84e9b

`xq_os_v1.0.4_mini_2017-11-17` can only be used on Xiaoqiang mini

[Download Address](#) md5: e874c78088211dff3feabd92f86275e7

Precautions:

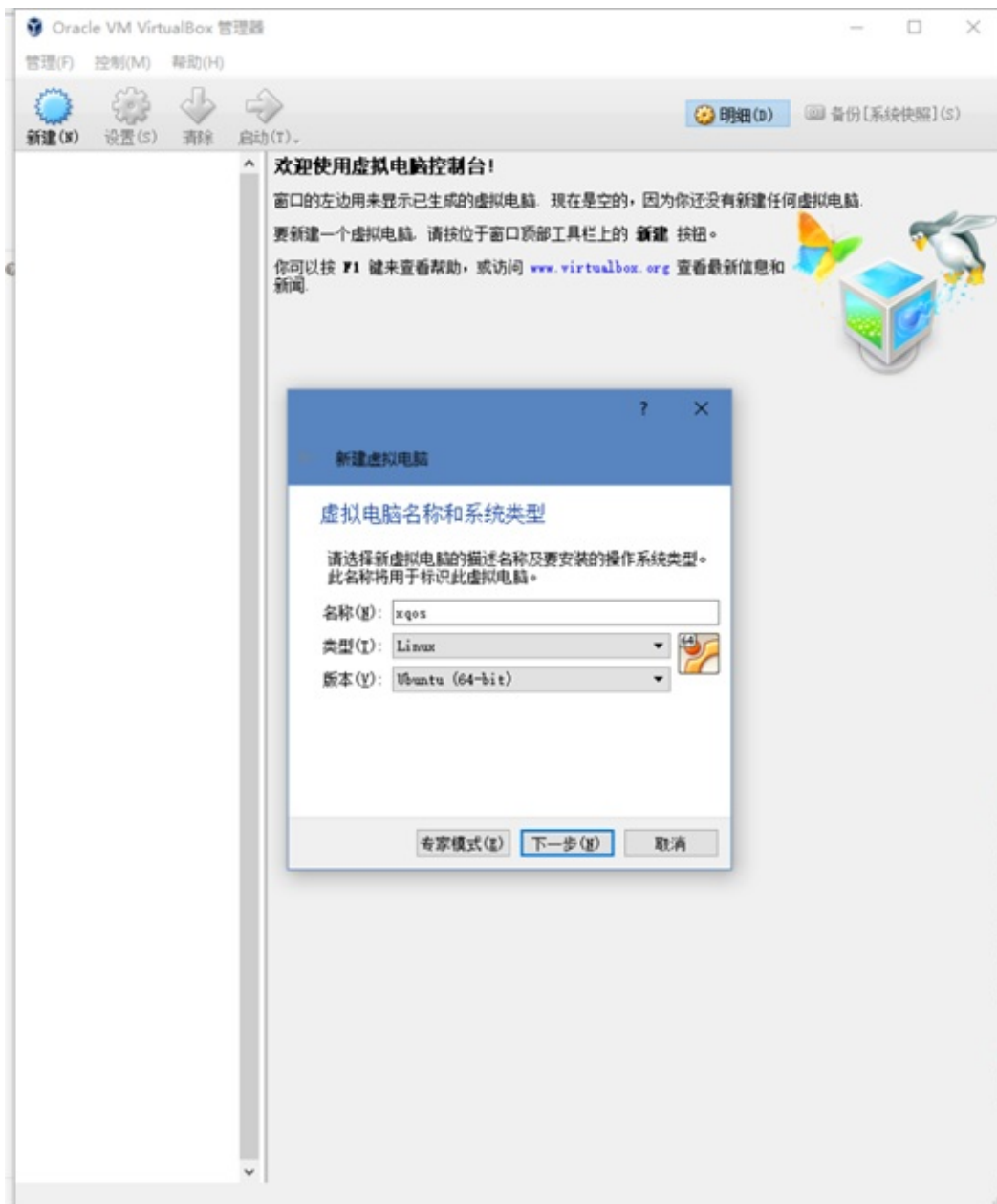
After the download is completed, you must check md5 to confirm the integrity of the file. Note that when installing the image, the user name can only be `xiaoqiang`, otherwise there will be problems.

Image installation method

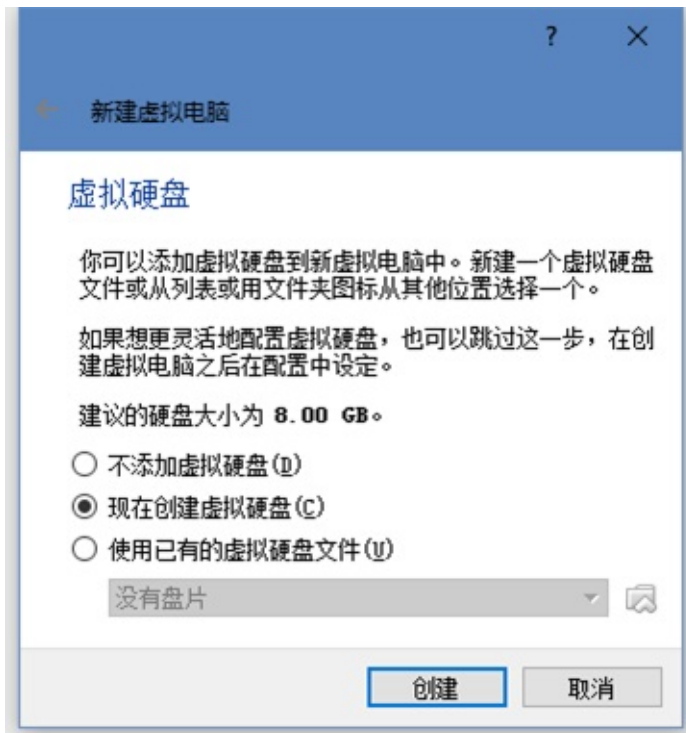
Install in a virtual machine

Open the virtual machine software. [Download link](#) Click the create button to start creating a virtual machine.

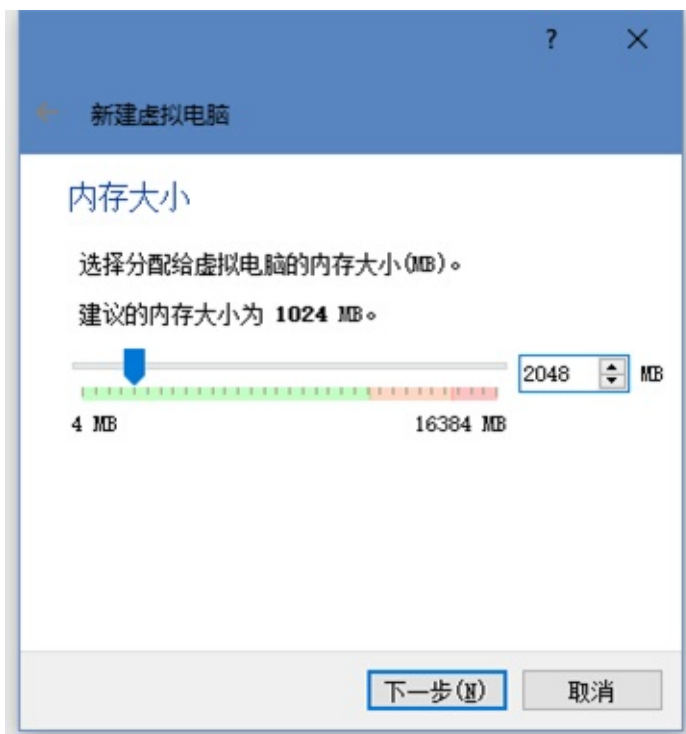
Note: please select `ubuntu 64` when select system.

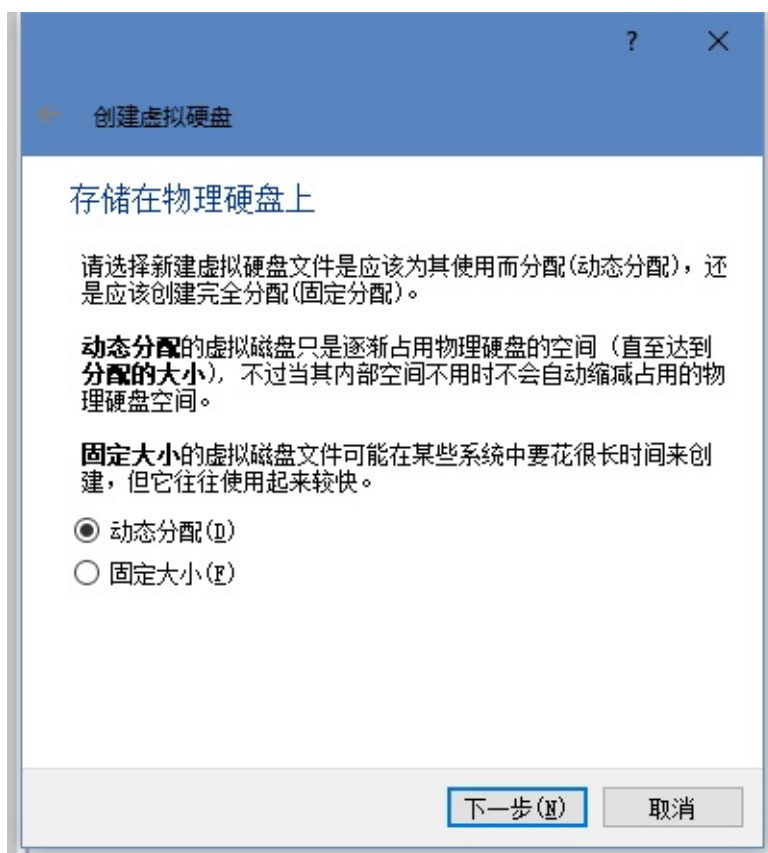
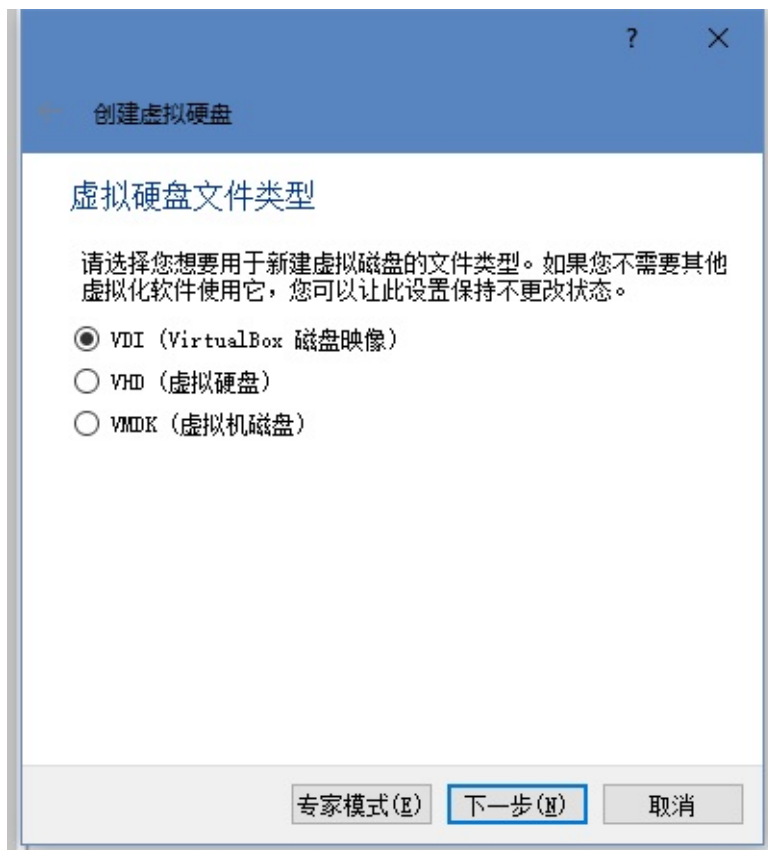


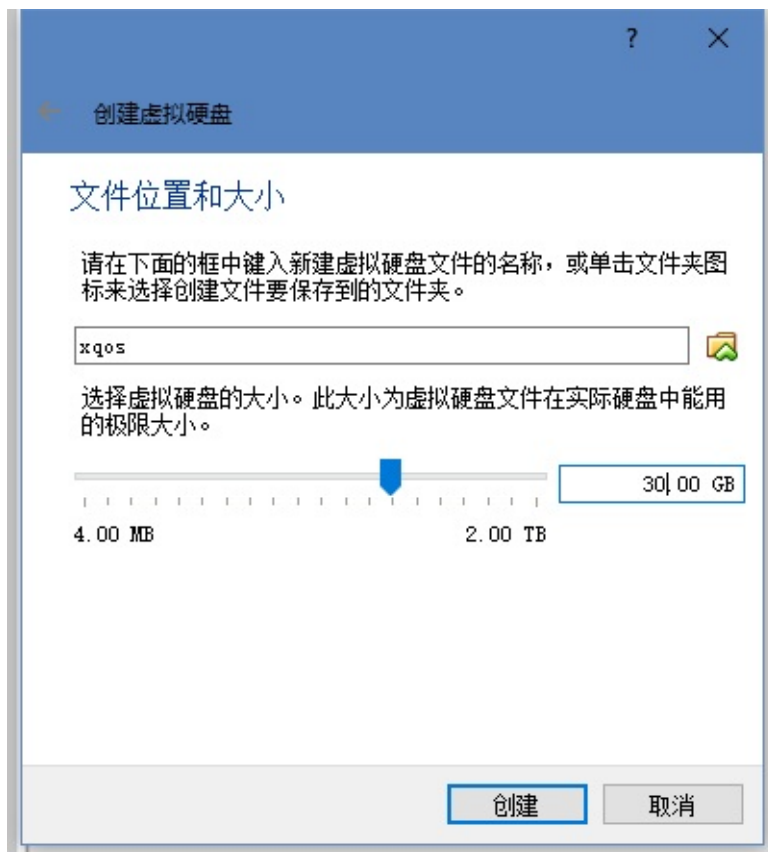
According to the settings on the image, click Next when finished.



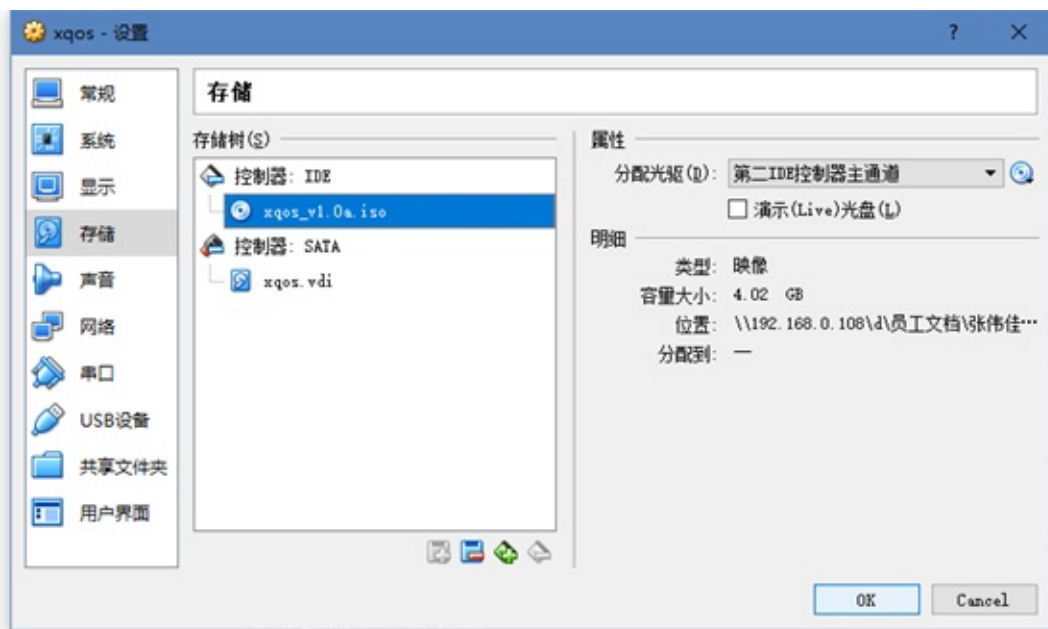
Memory is recommended to be set to 2G, if memory is too small, system may not able to start.



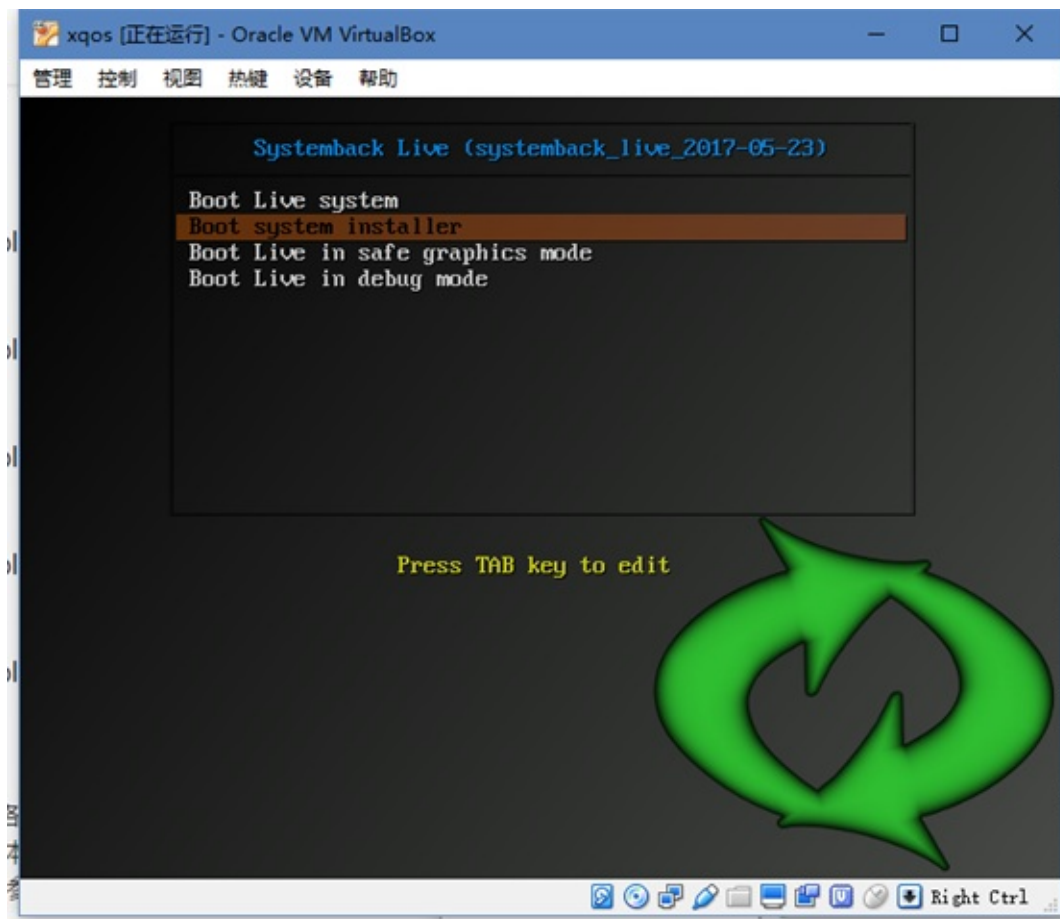




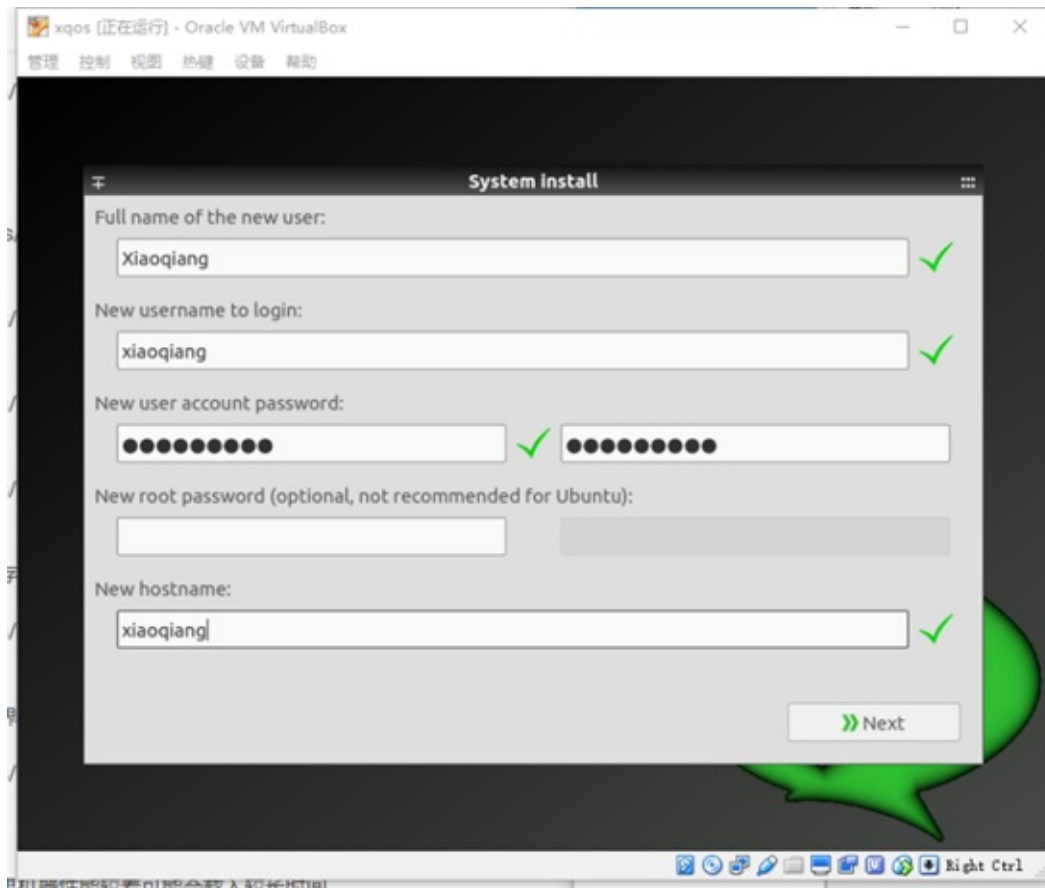
After setting is complete, click the Settings button above. In the popup dialog, select Storage. Then click the CD icon on the right of the storage interface to set the image file.



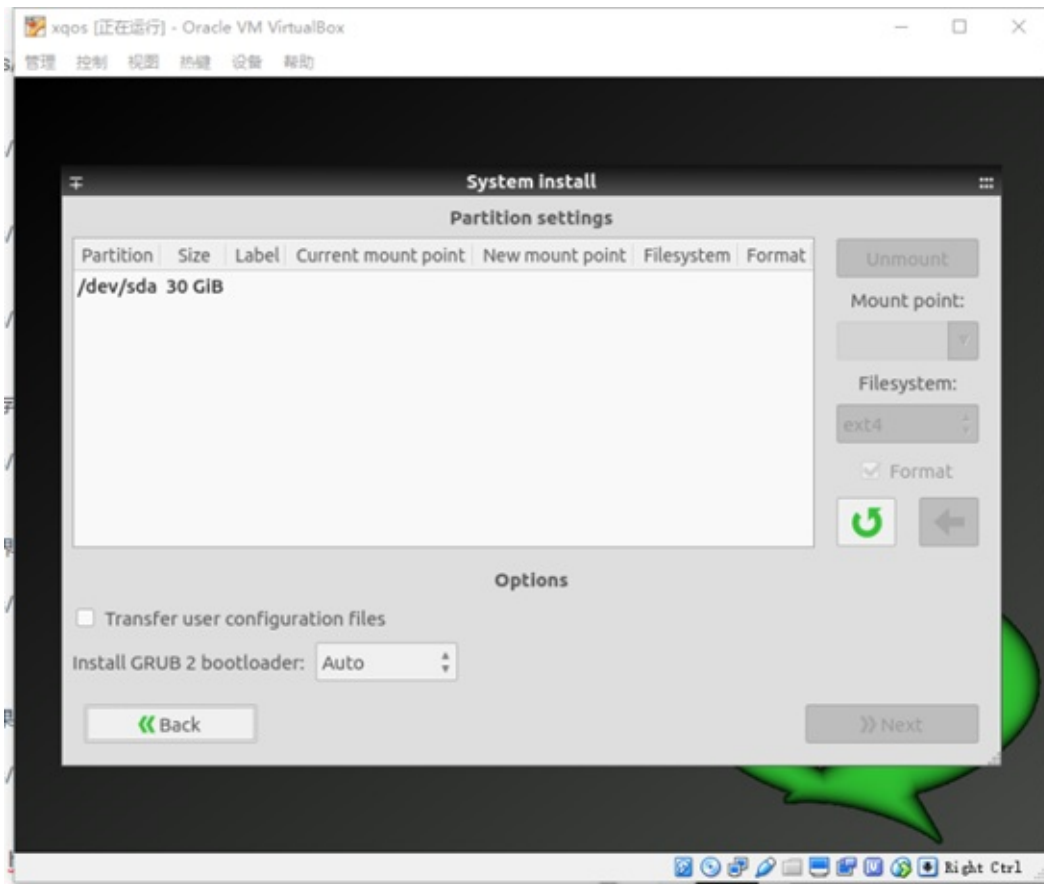
After the setting is completed, click the start button on the main screen. Waiting to load the system's selection interface.



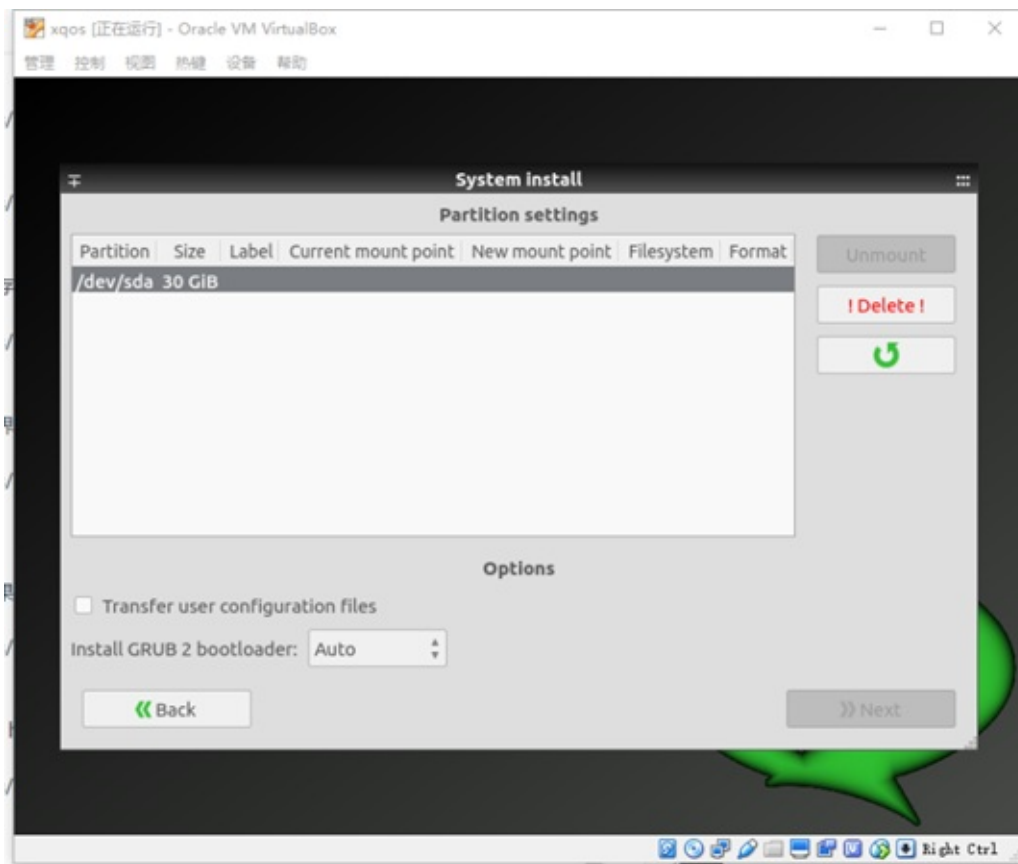
If you want to install the system, select the second option, then press Enter to confirm. Wait for the system installer to load. This process is related to your machine's performance. If the machine's performance is poor, it may take longer to load.



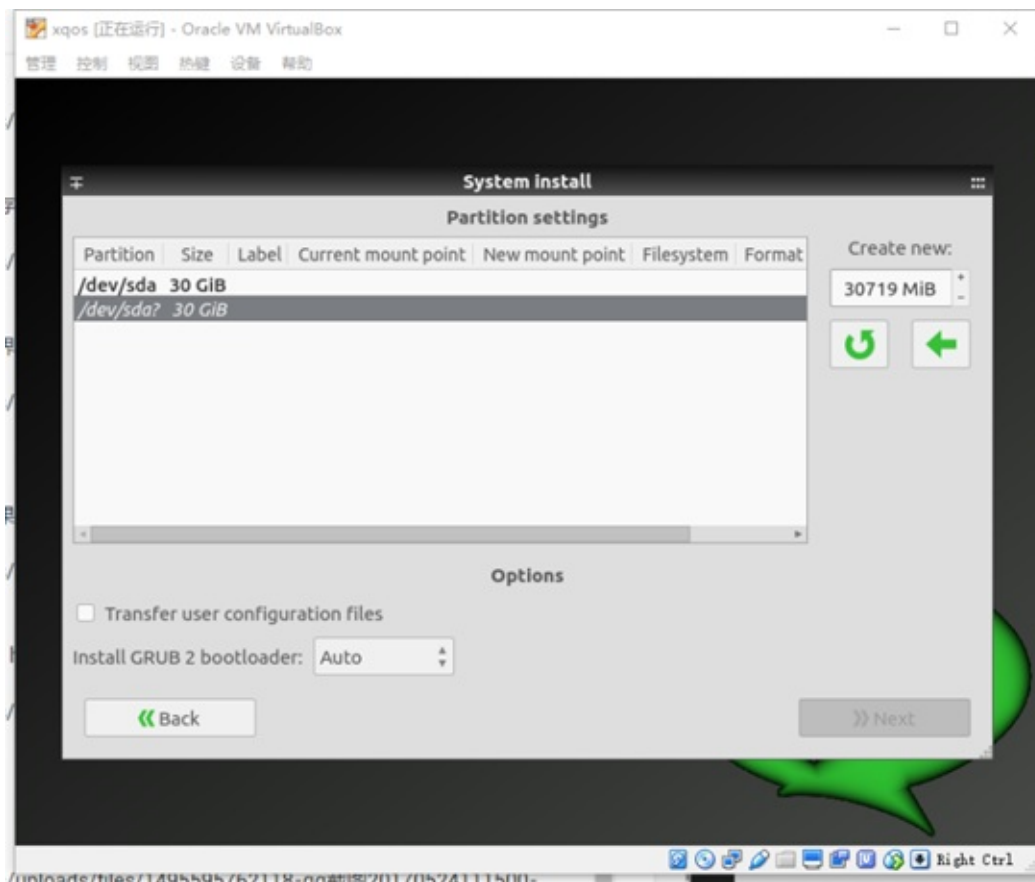
After the loading is complete, set the user information, note that the username can only be xiaoqiang, hostname can be set at will.



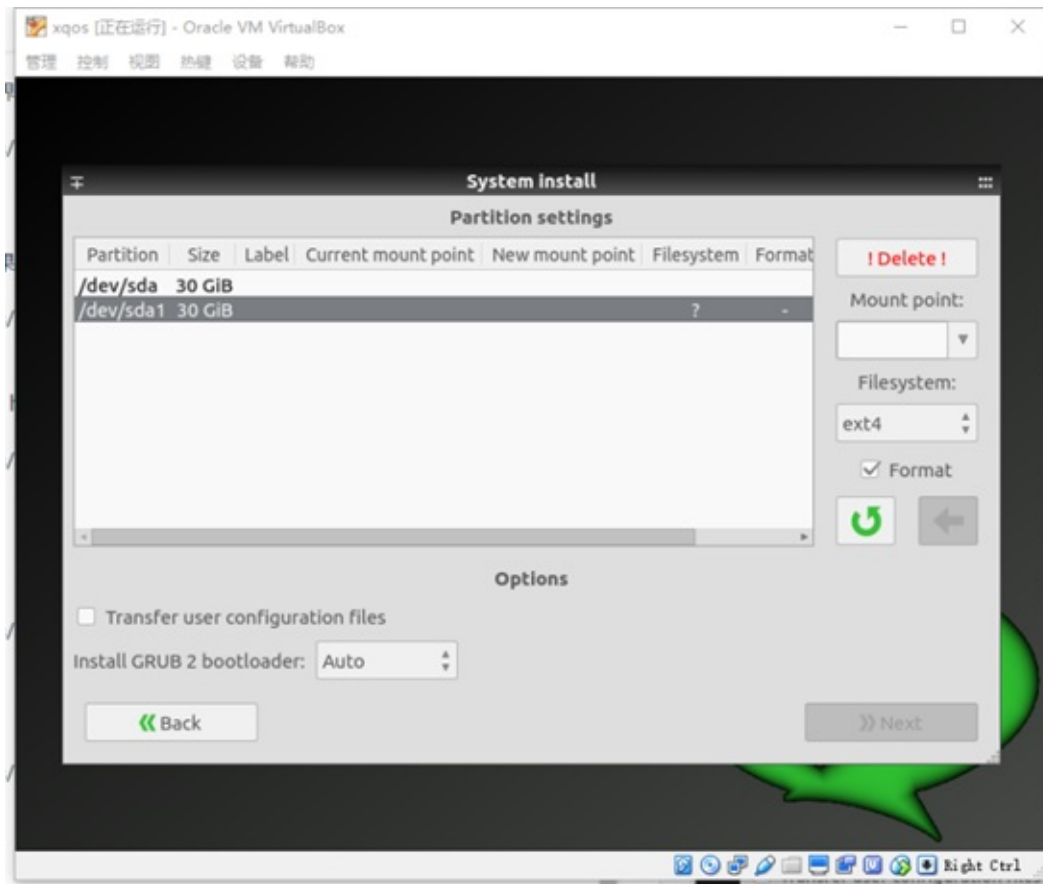
Click Next to enter the partition interface. Select the hard disk you want to install and click the Delete button.



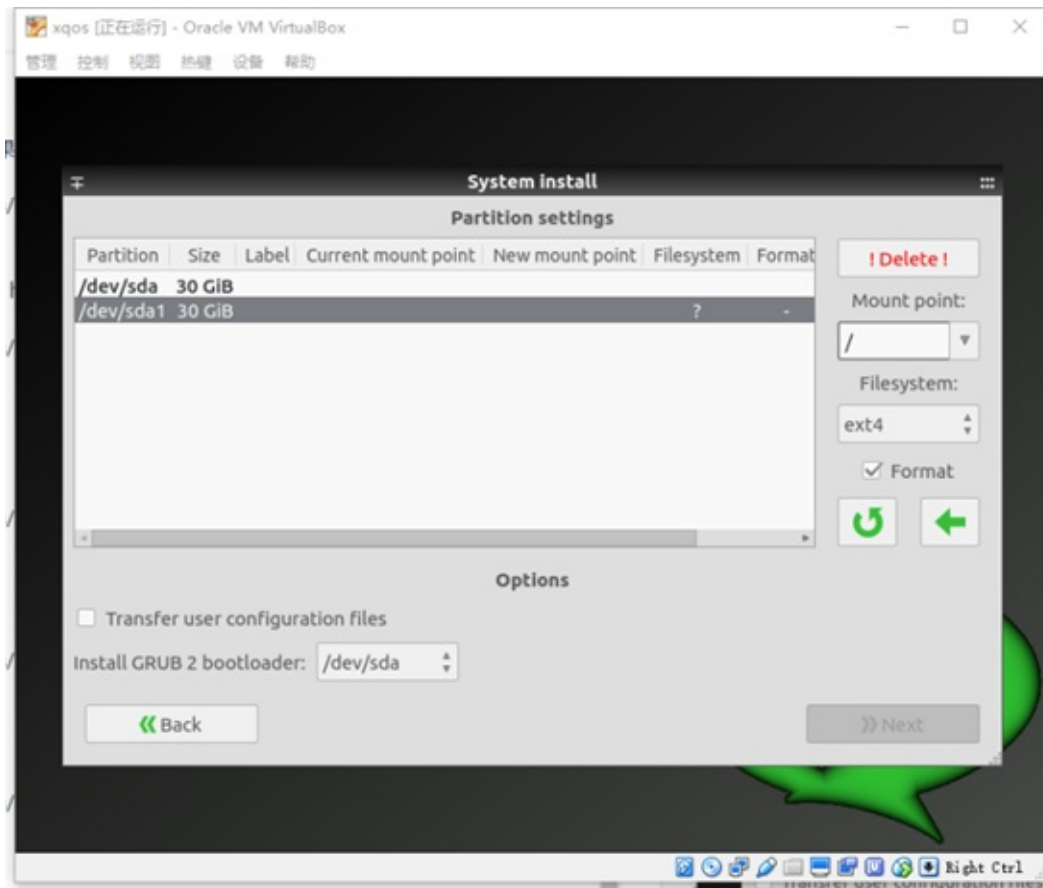
Select the new partition in the hard disk again.



Click the arrow to go to the next.

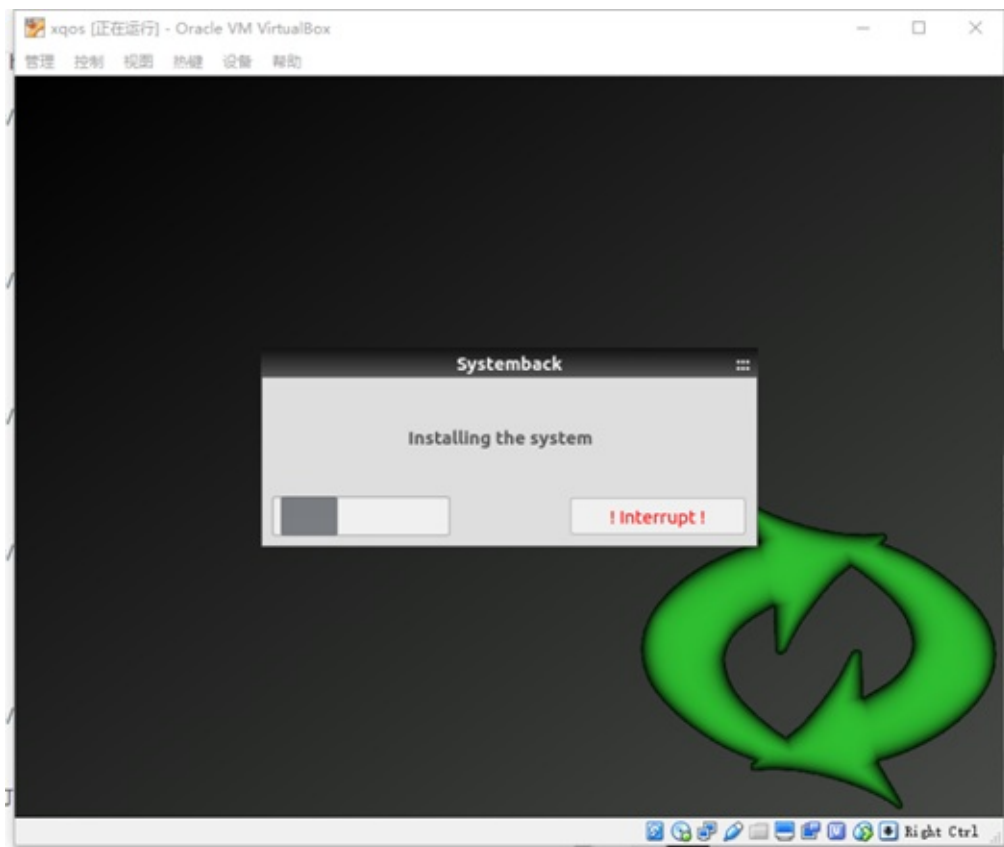


Select the newly created partition again, set the mount point on the right, click the arrow again to enter the next step.



Note: please check transfer user configuration files

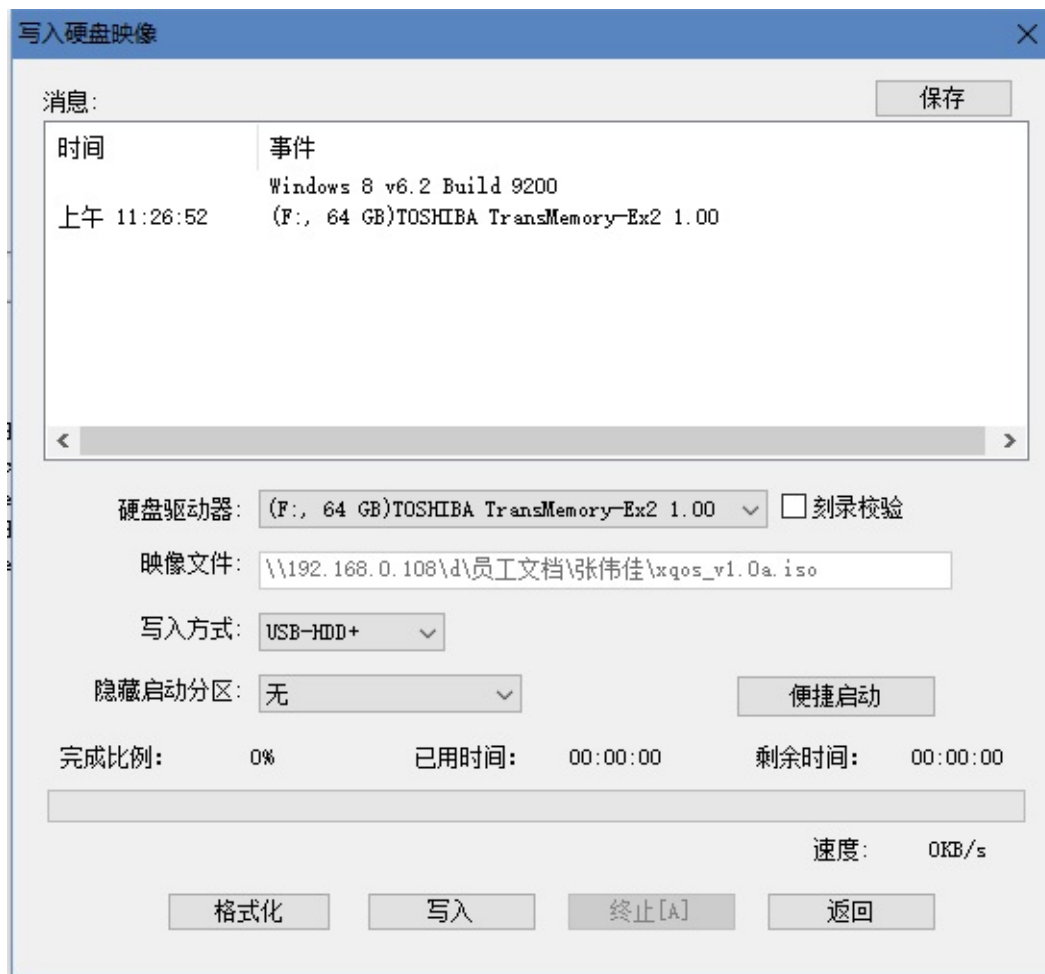
Click on the next lower right corner to continue. If you still want to set up other partitions, follow the above steps.



Wait for the system to restart after the installation is complete.

Install on actual computer

The installation process is generally the same as the installation of Ubuntu. Here's how to use USB disk to install the system. First you need to prepare a blank partition on your hard disk to install the system. Then download `ultroiso` . [Download address](#). Insert your USB disk into your computer. Open iso file with software. In the menu bar, select Start -> Write Disk Image, the popup interface is as follows.



Click Write and wait for the write to complete.

After writing, insert the USB disk into the computer where you want to install the system. Enter the BIOS option at startup and select Boot from USB drive. The subsequent installation process is basically the same.

Note that in the partition, set boot as a separate partition, and then select the grub installation location. Otherwise, the system cannot be started after the installation is complete. There are also differences between partitions for different boot modes. Start with grub to give /boot a separate partition, boot with uefi to give /boot/efi a separate partition

[Download The Virtual Machine](#)

- [How Ubuntu sets static IP](#)

How Ubuntu sets static IP

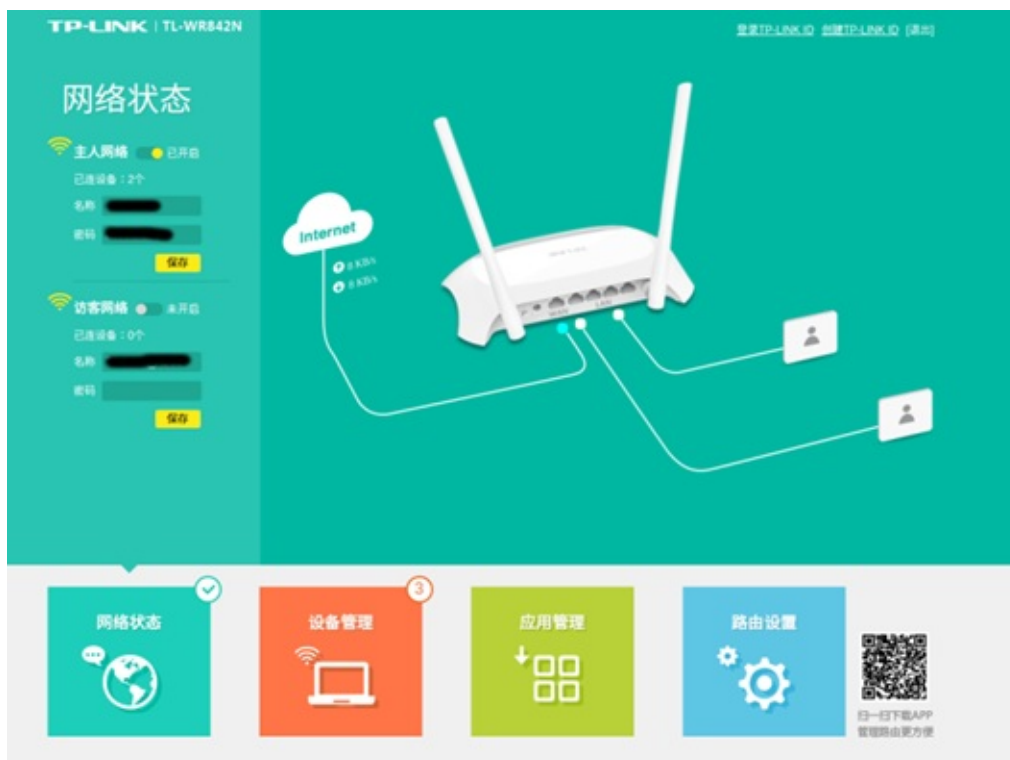
When searching for "Set Static IP" on the network, the results appear are set by modifying the file. This way it is easy to make the computer's network unusable. It is recommended to use the following method to modify.

The basic way of IP allocation

The way IP is allocated is generally determined by the router. The router has two modes: DHCP mode and static IP. DHCP is a dynamic way of assigning IP. The default router is this mode. In this mode, the computer can also set its own static IP. Of course, there is no guarantee that it will succeed. For example, the IP set by yourself may be occupied by others. It is also possible that the router will not assign you the IP you set because of the confusion. Let's talk about the various methods of setting static IP.

Several ways to set static IP

1. For router settings, setting up a static IP through a router is the easiest way. However, the premise is that you must have the management rights of the router, and the router should also support this setting function. First enter the address of the route in the browser and enter the management interface of the router.



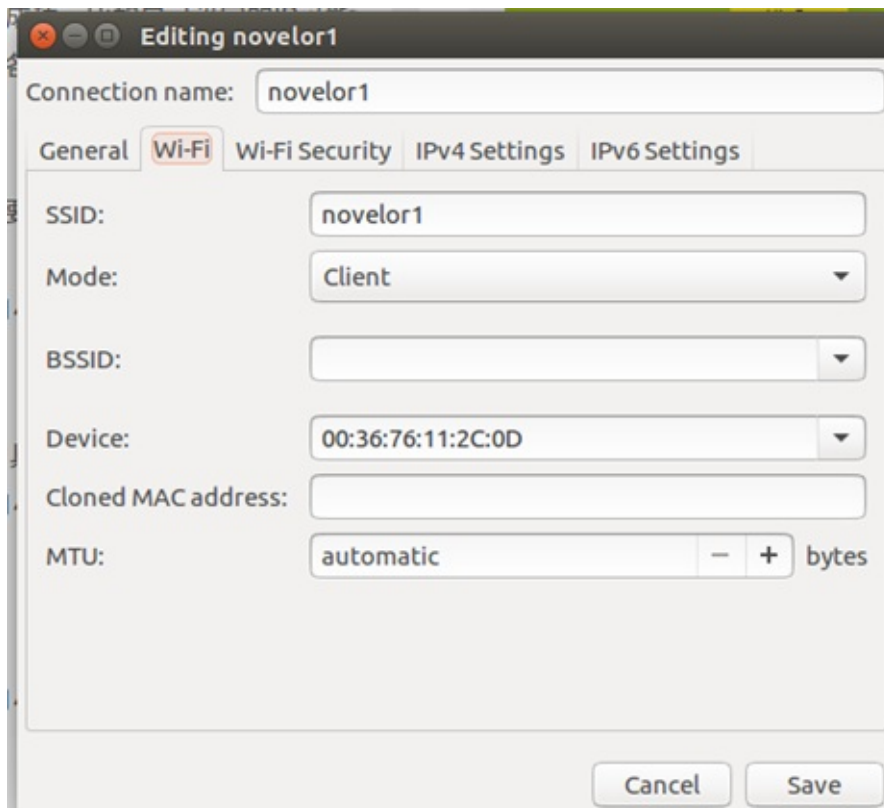
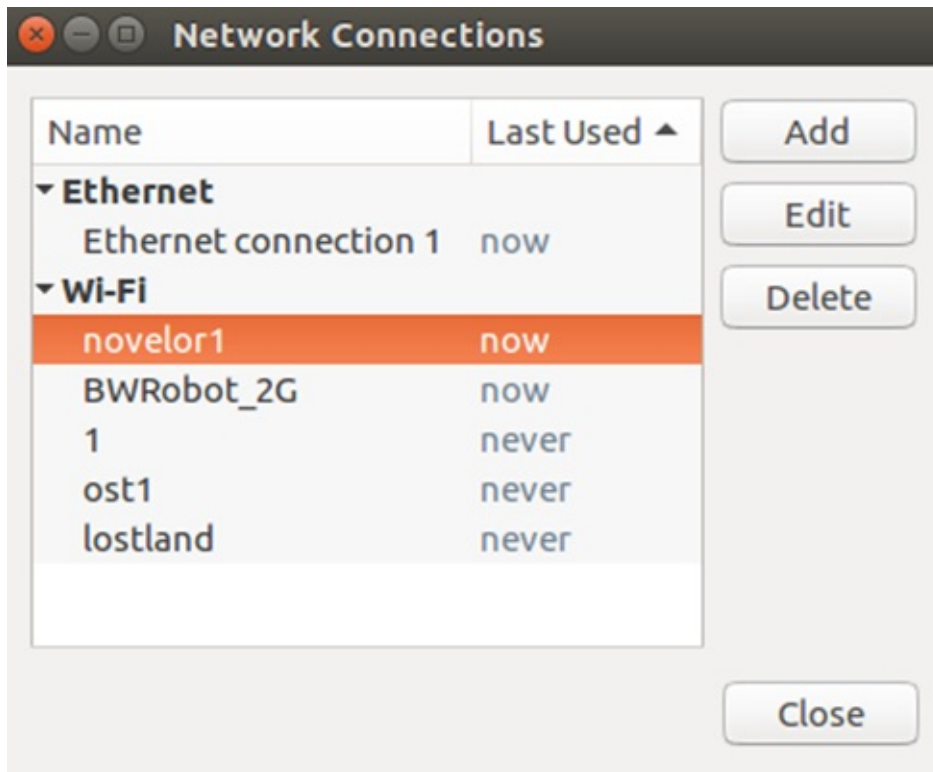
There are IP and MAC binding options in application management. Different router settings interface is different, you can find the specific location of this function.



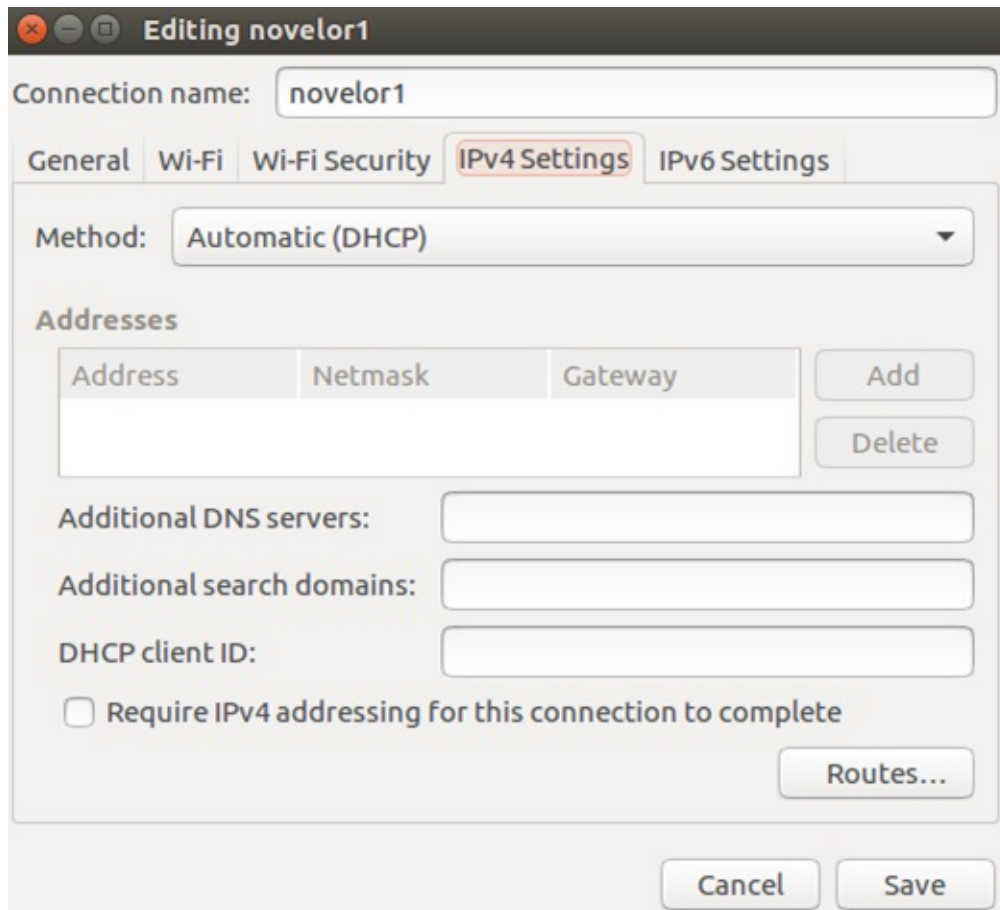
Here you can set up a static IP.



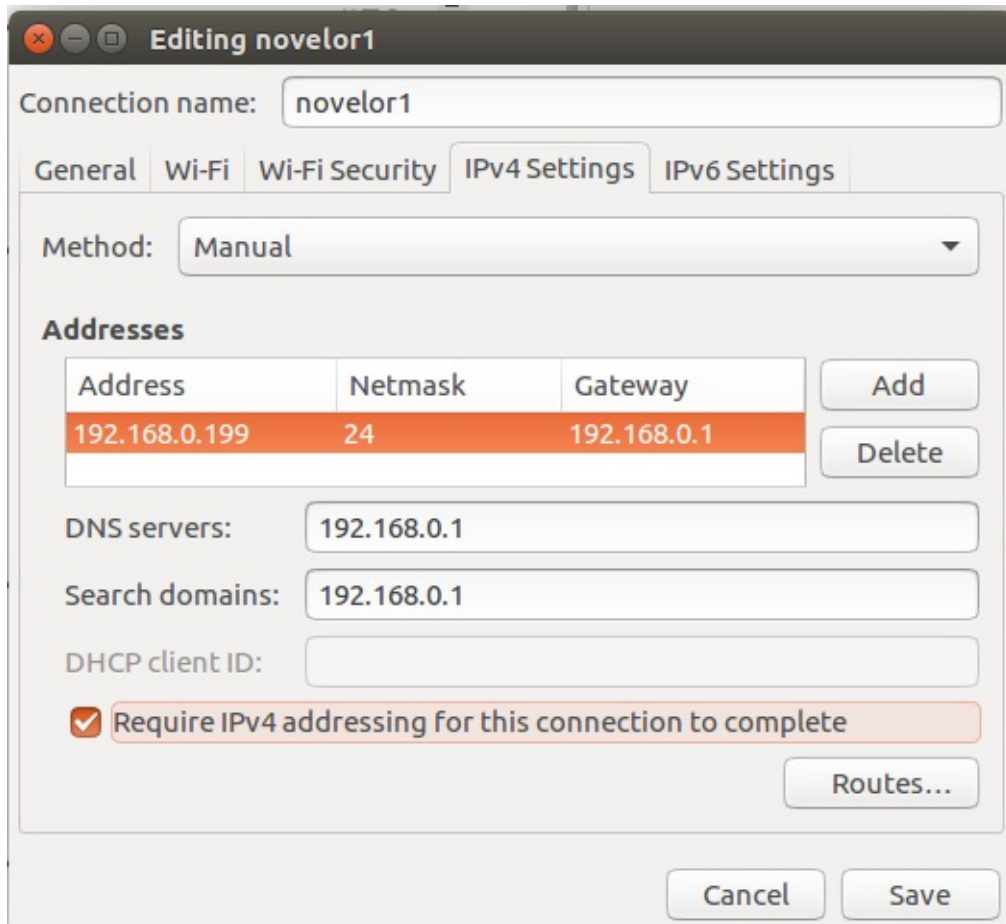
1. Make settings on the computer After the computer is connected to the network, open the computer's network manager and select your current network link, select Edit.



Click on IPv4 settings



Set up as shown below



192.168.0.199 is the local IP you want to set, so you can't repeat it with other IPs on the LAN. 192.168.0.1 is the address of the router. This should be adjusted according to your own network conditions. After the setting is completed, click Save to open, then disconnect the network and reconnect. It will take effect only after reconnecting.

```

andoms@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:feaf: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)

vlan1    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)

andoms@nowhere:~$

```

Enter `ifconfig` to view the current network information, you can find that my wireless network has been set to the IP just now.

- Xiaoqiang's remote assistance function
 - Execute htop at the terminal
 - Remote connect to Xiaoqiang
 - Remote connect to your xiaoqiang
 - Enable and disable remote assistance

Xiaoqiang's remote assistance function

In order to provide you with better service, Xiaoqiang, which was released after June 2016, has remote assistance software installed by default. Through this software our technicians can connect directly to your Xiaoqiang to solve technical problems for you. Of course, you can also use your remote assistance to control your own Xiaoqiang more conveniently. The following describes the specific use of Xiaoqiang's remote assistance function.

Confirm if remote assistance has been activated

Execute `htop` at the terminal

```
randoms@nowhere: ~/Downloads

 1  [|||||] 15.4%
 2  [|||||] 14.1%
 3  [||||]  8.0%
 4  [||||]  5.2%
Mem [|||||] 4828/15947MB
Swp [|||||]  0/0MB
 5  [|||||] 53.2%
 6  [||]  1.3%
 7  [||]  0.0%
 8  [||]  1.3%
Tasks: 228, 610 thr; 3 running
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
```

If you can see that the thread called `SharpLink` is running, then the remote assistance has already started. You can also confirm the status of the program and execute the instructions in a simpler way.

```
sudo service sharpink status
```

If the terminal displays:

```
xiaoqiang@xiaoqiang-desktop: ~
xiaoqiang@xiaoqiang-desktop:~$ sudo service sharplink status
[sudo] password for xiaoqiang:
● sharplink.service - Start sharplink service at startup
   Loaded: loaded (/lib/systemd/system/sharplink.service; enabled; vendor preset
   Active: active (running) since Tue 2018-09-25 15:12:37 CST; 22s ago
   Main PID: 2388 (cli)
   CGroup: /system.slice/sharplink.service
           └─2388 /usr/bin/cli /home/xiaoqiang/Documents/ros/devel/lib/sharplink

Sep 25 15:12:37 xiaoqiang-desktop systemd[1]: Started Start sharplink service at
Sep 25 15:12:37 xiaoqiang-desktop SharpLink.exe[2388]: ID: 4D87B61F3A66826911045
Sep 25 15:12:37 xiaoqiang-desktop SharpLink.exe[2388]: Server listening on 36063
Sep 25 15:12:53 xiaoqiang-desktop SharpLink.exe[2388]: From Server 36063:tox is
xiaoqiang@xiaoqiang-desktop:~$
```

This indicates that the program has been executed normally.

Remote connect to Xiaoqiang

Check your ID Each xiaoqiang has an unique ID, please don't tell others. Because if you don't change the default password and someone knows your ID, then he can easily manipulate your xiaoqiang.

The method of viewing the ID is also very simple, in the terminal execution

```
sudo service sharplink restart
bwgetid
```

and the output would be

```
D70101972AB9B7E674290A25485B3752EDA236F65EF2C4AC7E738390DA61903565E68B8C431B
```

This very long string is your ID.

If you would like us to provide remote assistance, just send this ID to our staff.

Remote connect to your xiaoqiang

After remembering your own ID, as long as your xiaoqiang has connected to the Internet, you can control it anywhere at any time, not by the LAN restrictions.

First install [Sharplink](#) on your PC, which is cross-platform, both Linux and Windows can be installed. The specific method of installation is in the introduction of the project.

After the installation is complete, the terminal executes

```
./SharpLink.exe 9999 [Your ID] 127.0.0.1 22
```

This command maps the local 9999 port to the 22 port of xiaoqiang. Just connect to the local 9999 port and you'll be connected to xiaoqiang's 22 port. Of course you can also change 9999 to your preferred port.

Now enter the following commands on local computer

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

Wait for the connection to complete, then you can control the xiaoqiang.

Enable and disable remote assistance

If you want to disable remote assistance

```
sudo systemctl disable sharplink
```

if you want to enable remote assistance

```
sudo systemctl enable sharplink
```

- [Google lidar slam algorithm Cartographer installation and bag package demo test](#)

Google lidar slam algorithm Cartographer installation and bag package demo test

Cartographer is a set of laser radar slam algorithm that was open sourced by Google in September 2016. Its precision and effect are among the best in the industry. This article will demonstrate how to use the ROS JADE version. First go to the tutorial demo video click to watch



Steps:

1. Installation dependency package

```
# 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. Install 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. Install cartographer

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

4. Install cartographer_ros

```
cd ~/Documents/ros/src # Please modify the path to your own ROS catkin workspace
git clone https://github.com/BlueWhaleRobot/cartographer_ros.git
cd ..
catkin_make
```

5. The installation is complete, start downloading the bag file for testing

Click the link below to download the file and save it to your desktop.

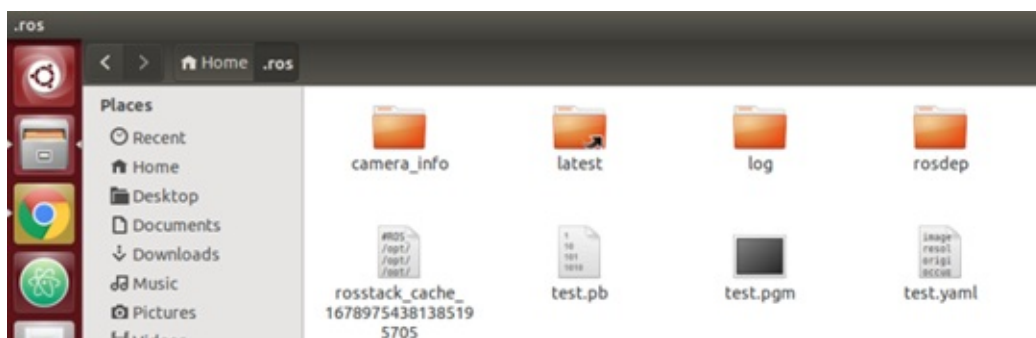
6. Start the demo, you can see rviz start and start to build

According to the computing power of the personal platform, the complete running time of this demo is generally between half an hour and one hour.

```
roslaunch cartographer_ros demo_backpack_2d.launch bag_filename:=${HOME}/Desktop/cartographer_paper_deutsches_museum.bag
```

7. Save the map and end the test

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



The created map file will now be generated in the `~/ros` folder in the home directory. These two files (`pgm` and `yml`) can be loaded in the `map_server` in `ros`.

- [Install VNC on Ubuntu](#)
 - [Installing X11VNC](#)
 - [Set access password](#)
 - [Create a service file](#)
 - [Start the service](#)
 - [Access the service](#)

Install VNC on Ubuntu

When using Ubuntu often requires Remote Desktop connection, the most common software is VNC. VNC is an open protocol that implements a lot of clients. But after comparing the various implementations, the best is now x11vnc. This program is not only free of charge, open source, but also supports OpenGL programs. For example, rviz and other programs can also open normally.

The X11VNC is already installed by default in the xiaoqiang system image, and if you use a xiaoqiang host or mirror, you can skip the installation process directly. Just follow the instructions for accessing the service.

The installation method is described below. The following is an example of Ubuntu 14.04, if it is 16.04 and later version, you need to edit the service configuration file.

Installing X11VNC

```
sudo apt-get install x11vnc -y
```

Set access password

```
sudo x11vnc -storepasswd /etc/x11vnc.pass
```

Create a service file

Under/etc/init, create a x11vnc.conf file with the following file contents

```
description "xiaoqiang vnc server"
start on runlevel [2345]
stop on runlevel [06]

script
    exec /usr/bin/x11vnc -auth guess -capslock -forever -loop -noxdamage -repeat -rfbauth
    /etc/x11vnc.pass -rfbport 5900 -shared
end script
```

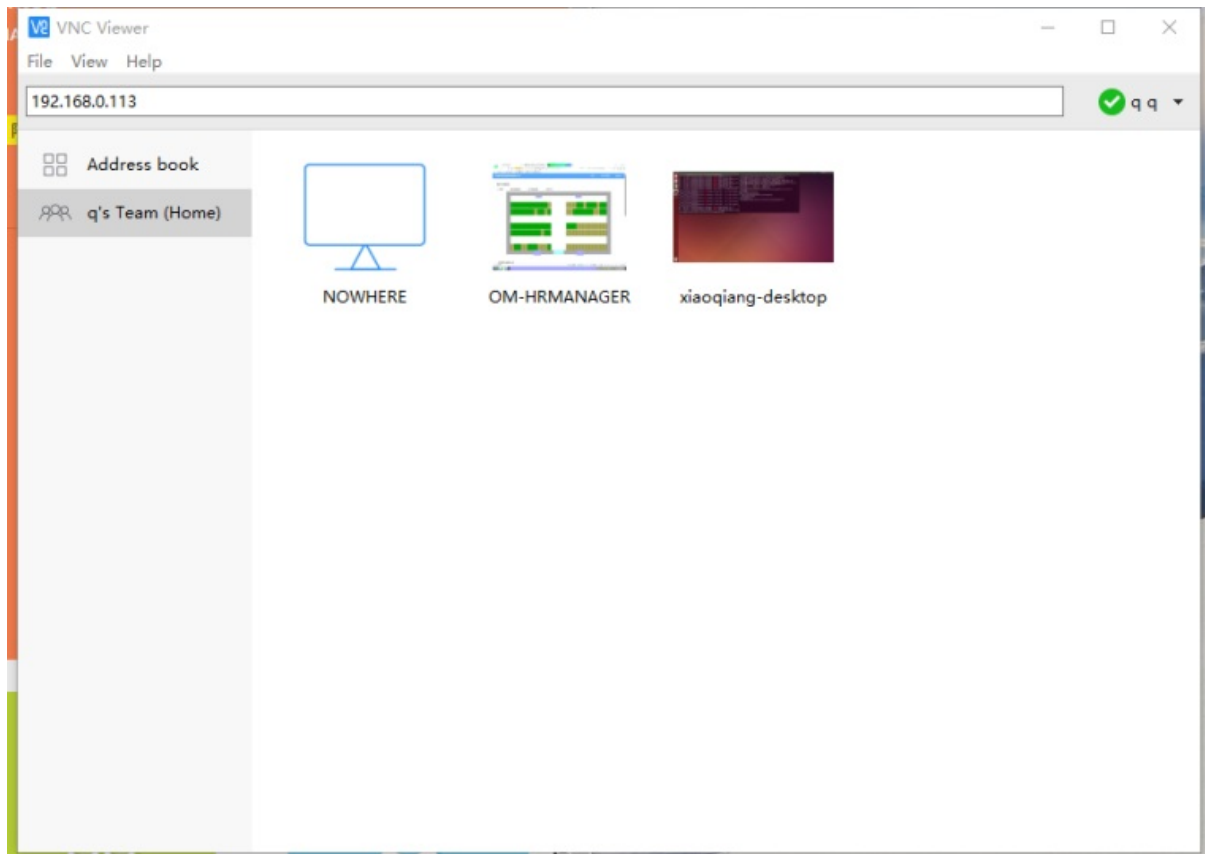
Start the service

```
sudo service x11vnc start
```

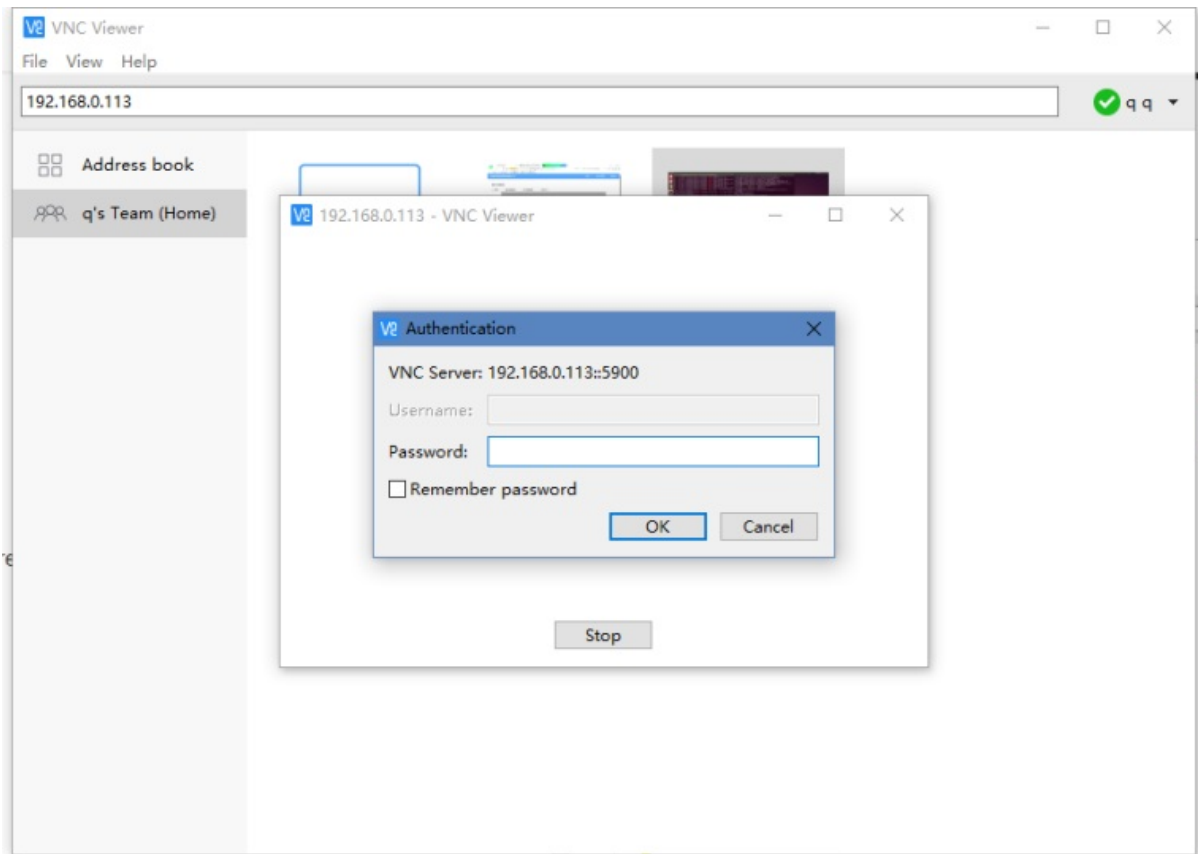
Access the service

Download a VNC client, such as download from [here](#)

Start the client, enter target ip address



and then enter access password



Then you can connect normally.

For Ubuntu 16.04 and later, You can configure your service file according to this [script](#)

Note when the installation is complete, you may still get an error when you use `rviz` without plugging in the monitor. Just plug in the HDMI to VGA adapter (do not connect the monitor, just the adapter) can be turned on normal use. If you want to adjust the resolution in the same way as normal computer settings, adjust the settings inside system settings

- [PS3 joystick ROS driver](#)
 - [installation steps](#)
 - [Quick usage](#)

PS3 joystick ROS driver

[Xiaoqiang Homepage](#)

The ros driver of the ps3 controller is ps3joy in the joystick_drivers package. This driver supports the original Sony handle, but there is a problem with the support of some others ps3 controllers. We have modified on the basis of ps3joy, added a `ps3joyfake_node.py` script as a driver for other joysticks, the package source code address is [here](#). Let's take Xiaoqiang as an example to demonstrate the installation steps and simple usage of this package.

installation steps

ssh enters Xiaoqiang ros workspace, compile after downloading the source code. If you have install xiaoqiang system image, you can skip this step.

```
ssh xiaoqiang@192.168.xxx.xxx
cd Documents/ros/src/
git clone https://github.com/BlueWhaleRobot/joystick_drivers.git
cd ..
catkin_make
If the following error is prompted
error spnav.h no such file
First install the following package, then re-execute catkin_make
sudo apt-get install libspnav-dev
```

Quick usage

After `ps3joyfake_node.py` starts, it converts the data of the joystick button received by bluetooth into the standard joy msg, and publishes it in ros with `/joy` as the topic, that is, a file of `ps3joyfake_node.py` is equivalent to two files of `ps3joy.py + joy_node`. It is no longer necessary to open the `joy_node` node in actual use.

1. Bind the joystick to the usb bluetooth adapter, just bind once, skip this step next time.

Connect the joystick to the host through the usb data cable, and plug the USB bluetooth adapter into the host.

```
sudo bash
roslaunch ps3joy sixpair
```

At this point, you will get an output similar to following. The current and setting mac addresses are the same.

```
Current Bluetooth master: 00:22:b0:d0:5a:09
Setting master bd_addr to 00:22:b0:d0:5a:09

# If the following error occurs
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.
# Run
hciconfig hci0 reset
# If you run hciconfig hci0 reset error
# Can't init device hci0: Operation not possible due to
# Run
rfkill unblock all
# and run
hciconfig hci0 reset
# rerun
rosvun ps3joy sixpair
```

After the binding setting is completed, disconnect the handle and the usb connection of the host.

Ctrl+D exits root mode

2. Pair the handle with the usb Bluetooth adapter

```
# Make sure the Bluetooth receiver is plugged into the host usb port
sudo bash
rosvun ps3joy ps3joyfake_node.py
```

The following prompt will appear normally.

```
root@xiaoqiang-desktop:~# rosvun 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.
If the following error is prompted
ImportError: No module named bluetooth
Please install the following packages first, then re-run
sudo apt-get install libbluetooth-dev
sudo pip install PyBluez
```

Press the joystick pairing button in the image below



If the pairing is successful, the above window will output a result similar to the following.

```
root@xiaoqiang-desktop:~# rosrun 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
```

3. View handle output

```
# Open a new window and print the button output
rostopic echo /joy
```

Normally the following similar results will appear

```
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, 0]
```

As we can see that the fourth button was pressed.

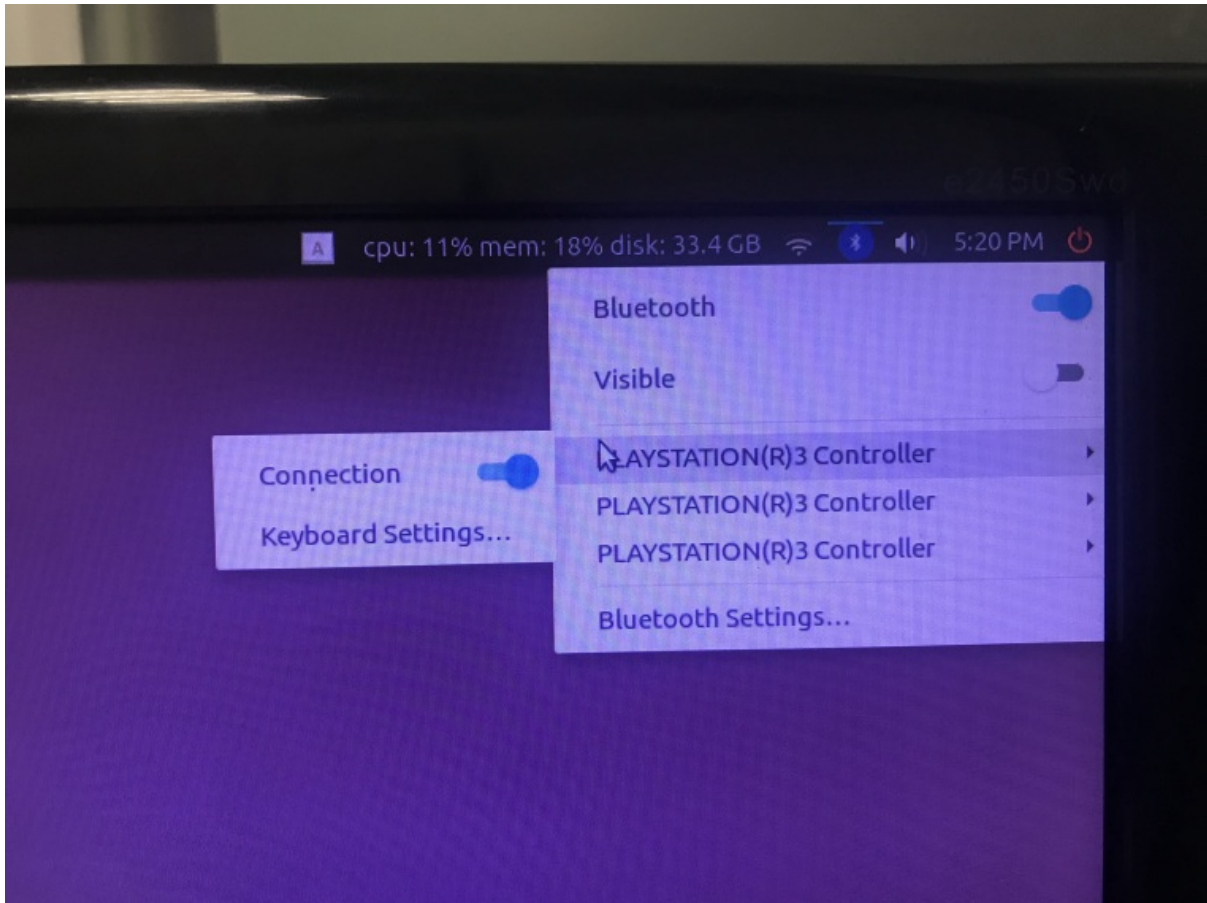
4. Start the relevant joy msg processing node

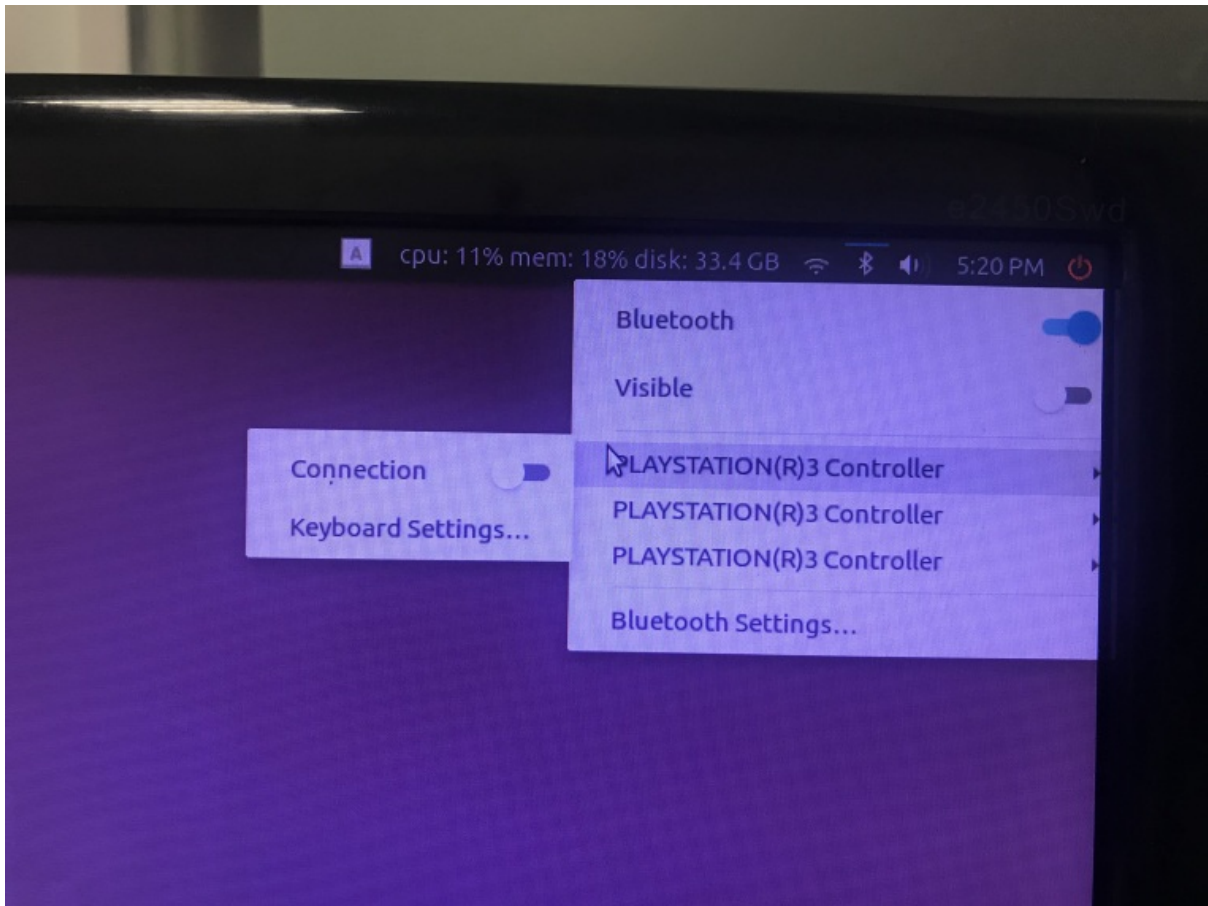
Be careful not to start the joy_node again.

```
# Take Xiaoqiang as an example. After launching the following launch file, you can remotely move Xiaoqiang.
roslaunch turtlebot_teleop ps3fakexiaoqiang_teleop.launch
```

For xiaoqiang user with a joystick you can also use the joystick according to this [tutorial](#)

When step 1 is bound, if the prompt cannot find the handle device and cannot be bound, you can perform the following two steps to solve the problem: 1. Click the Bluetooth icon in the upper right corner of the system desktop to close the connected Bluetooth handle service.





[Xiaoqiang Homepage Back To Index](#)

- [Cartographer install and demo](#)
 - [Steps:](#)

Cartographer install and demo

[Xiaoqiang Homepage](#)

[Cartographer](#) is a set of laser radar slam algorithm that was [open sourced](#) by Google in September 2016. Its precision and effect are among the best in the industry. This article will demonstrate how to use Cartographer in the ROS Kinetic. Click to view the result video.



Steps:

1. Installation dependency package

```
# 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. Install 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. Install protobuf 3.0

```
cd ~/Documents
git clone https://github.com/google/protobuf.git
cd protobuf
git checkout v3.6.1
mkdir build
cd build
cmake \
  -DCMAKE_POSITION_INDEPENDENT_CODE=ON \
  -DCMAKE_BUILD_TYPE=Release \
  -Dprotobuf_BUILD_TESTS=OFF \
  ../cmake
make -j 2
sudo make install
```

4. Install cartographer

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

5. Install 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
```

6. The installation is complete, start downloading the bag file for testing

Click the [link](#) to download the file and save it to your desktop.

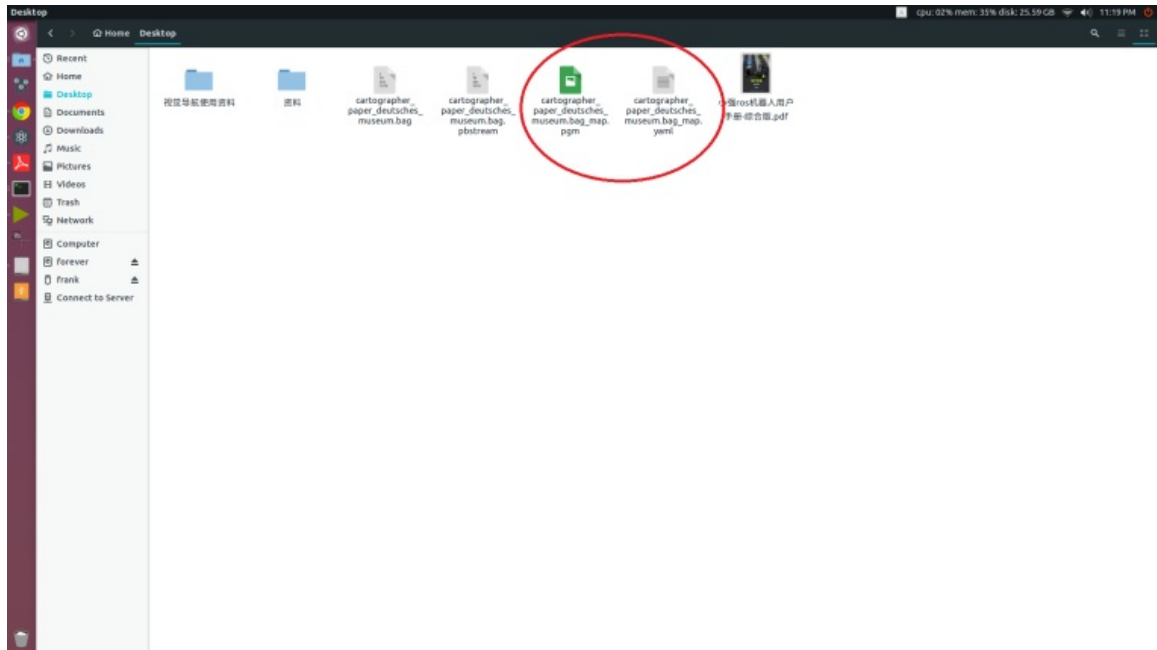
7. Start the demo, you can see rviz start and start to build

According to the computing power of the personal platform, the complete running time of this demo is generally between half an hour and one hour.

```
roslaunch cartographer_ros offline_backpack_2d.launch bag_filenames:=${HOME}/Desktop/cartographer_paper_deutsches_museum.bag
```

8. Save the map and end the test

```
roslaunch cartographer_ros assets_writer_ros_map.launch bag_filenames:=${HOME}/Desktop/cartographer_paper_deutsches_museum.bag pose_graph_filename:=${HOME}/Desktop/cartographer_paper_deutsches_museum.bag.pbstream
```



The created map file will now be generated in the .ros folder in the home directory. These two files (pgm and yaml) can be loaded in the map_server in ros.

[Xiaoqiang Homepage](#) [Back To Index](#)

- [Visual Navigation Path Editor Tutorial](#)
 - [Installation](#)
 - [Collecting spatial data](#)
 - [Launch software](#)
 - [Import Data](#)
 - [Draw a navigation route](#)
 - [Set navigation key points](#)
 - [Export data](#)

Visual Navigation Path Editor Tutorial

[Xiaoqiang Homepage](#)

Using Xiaoqiang can create a three-dimensional map of the surrounding environment, but how to use this map to visual tracking? The Visual Navigation Path Editor was written to implement this feature. With this tool you can mark the trajectory in map. Then export the generated track file to Xiaoqiang, and Xiaoqiang will be able to move according to the track you have drawn. The following describes in detail how to use the software.

Installation

Ubuntu deb installation package can be downloaded [here](#) and the source code is [here](#).

After the download is complete, execute the following command to install.

Note that this path editor is not compatible with the new version of the navigation program. For the new version of the navigation program, please use the [Windows client](#)

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

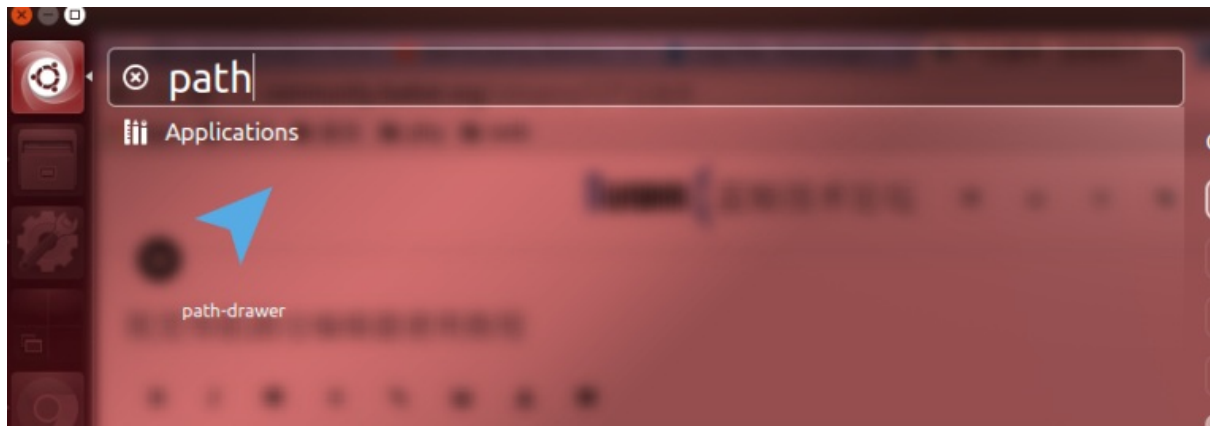
Wait for the installation to complete.

Collecting spatial data

The path editor needs to load the spatial data collected by Xiaoqiang to be able to operate. For detailed operations, please refer to [this article](#). After clicking the Save button, the map information will be saved to the `/home/xiaoqiang/slamdb` folder.

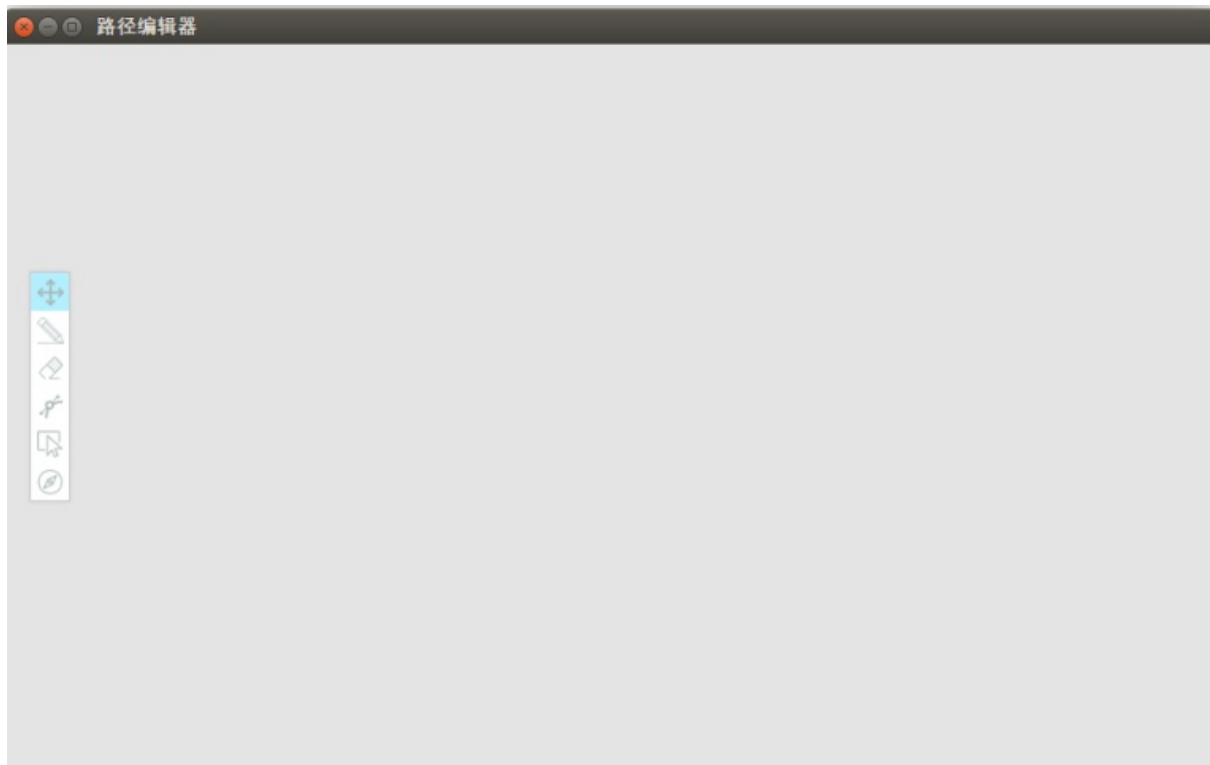
Launch software

After the installation is complete, you can find the program named Path Drawer in the Ubuntu Dash menu, and click on the software icon.



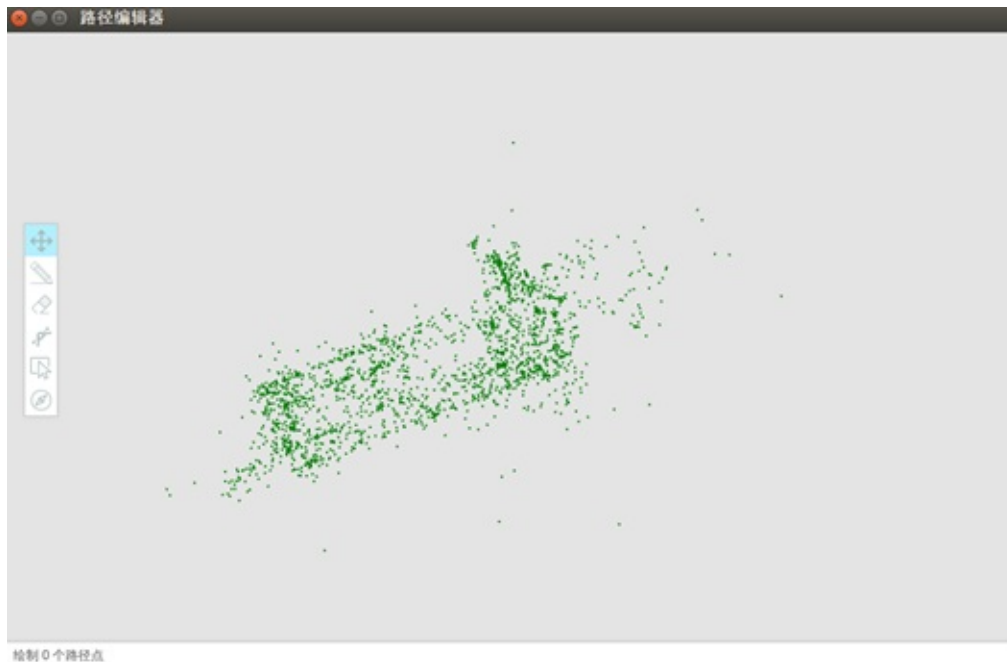
Import Data

The software interface after startup is shown as below

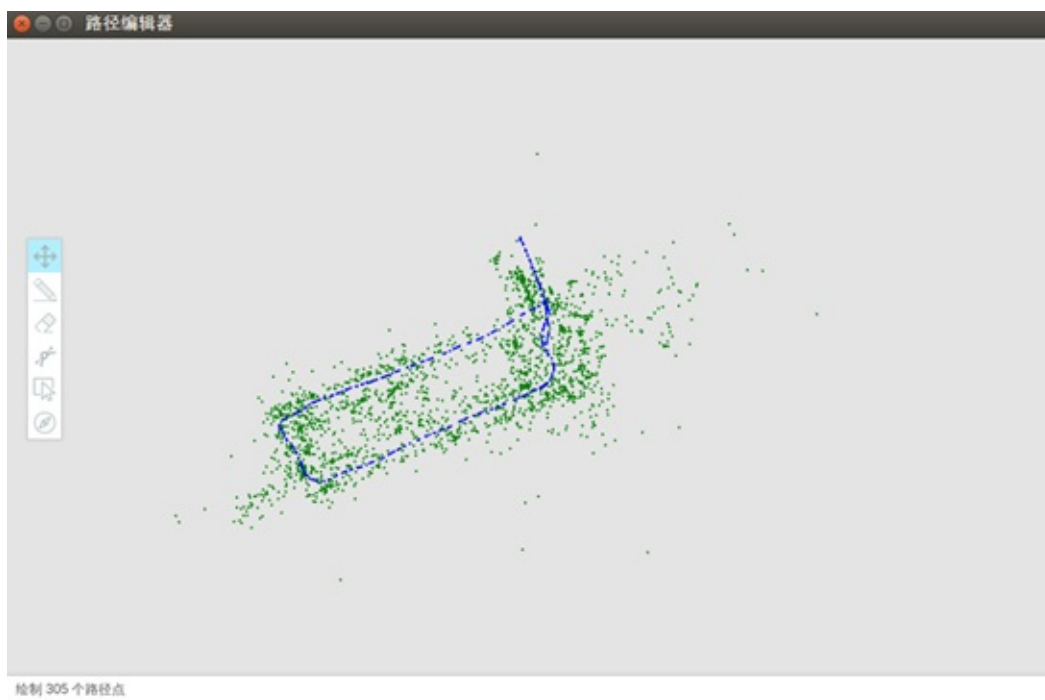


状态: 编辑中

In the menu at the top left, select File -> Import Map Data. In the file selection dialog that pops up, select the `/home/xiaoqiang/slamdb/mappoints.bson` file. After import succeed, you can see the data of the map points in the software. This is a top view.



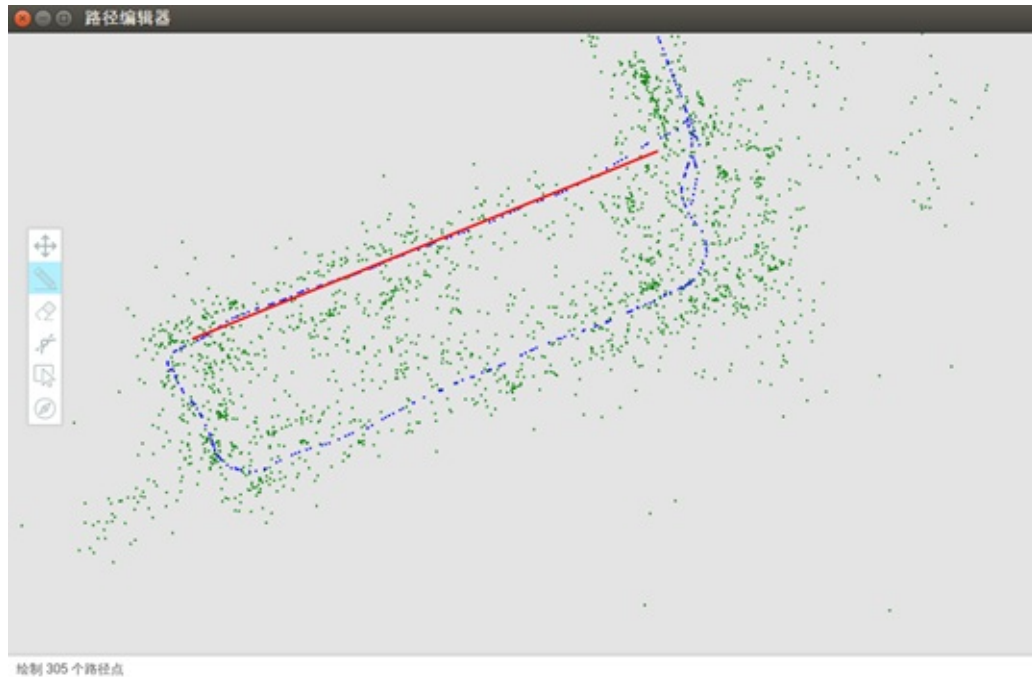
Then continue to select the file in the menu in the upper left corner and then import the path file. In the file selection dialog that pops up, select the `/home/xiaoqiang/slamdb/keyframes.bson` file. After import succeed, you can see the path of the previous car walking in the software.



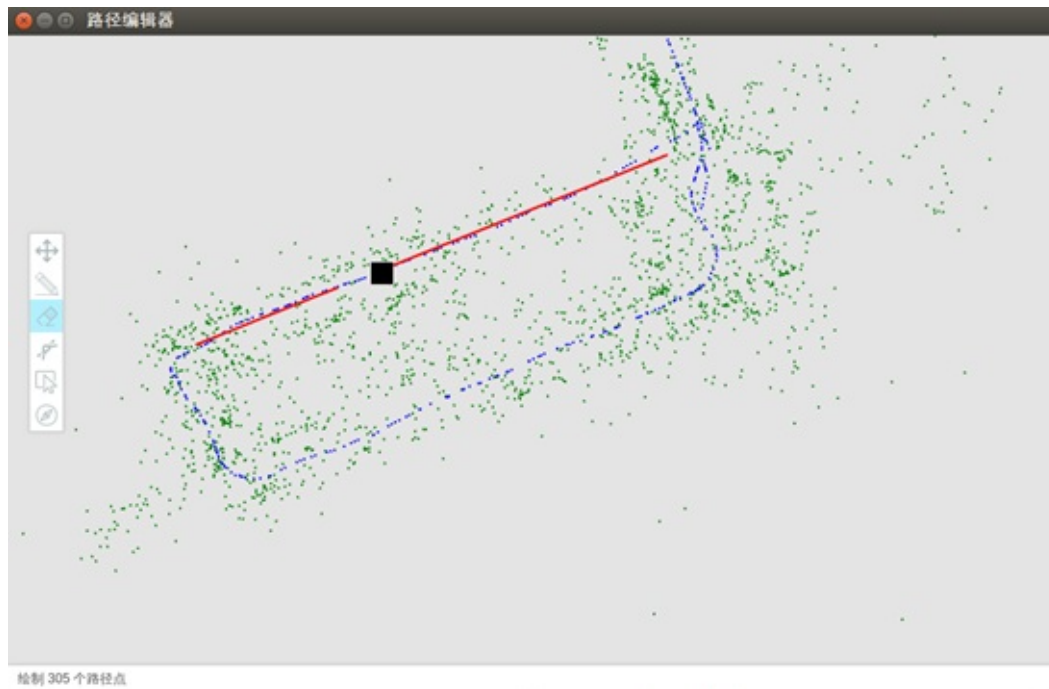
Draw a navigation route

The navigation path is the path you want xiaoqiang to walk. When the data is exported to Xiaoqiang, Xiaoqiang will move according to the path you draw. Here's how to use the path drawing tool.

1. Basic operations Basic operations include panning and zooming. If you drag the map with the left mouse button, you can the translation of the map. The scrolling of the mouse scrolls can zooming of the map, which is very useful in the process of drawing paths. For more detailed requirements on the movement, you can zoom in and draw.
2. The line tool: clicks the pencil-like icon in the left-hand toolbar. This is the line tool. left mouse button click on any point on the graph, and then move the mouse will appear a red line. Move the mouse to the desired end position, click the left mouse button again, a straight line to draw the finished. After you click the left button, you can cancel the drawing by right-clicking.



3. The eraser tool: Click on the Eraser tool in the left-hand toolbar and then presses the left mouse button to drag to erase the previously drawn point



4. Curve tool: Click on the left Curve tool, click the left mouse button at the beginning of the curve, then click the left mouse button again in the middle of the curve, and finally click on the end point of the curve. So a curve is drawn and done.

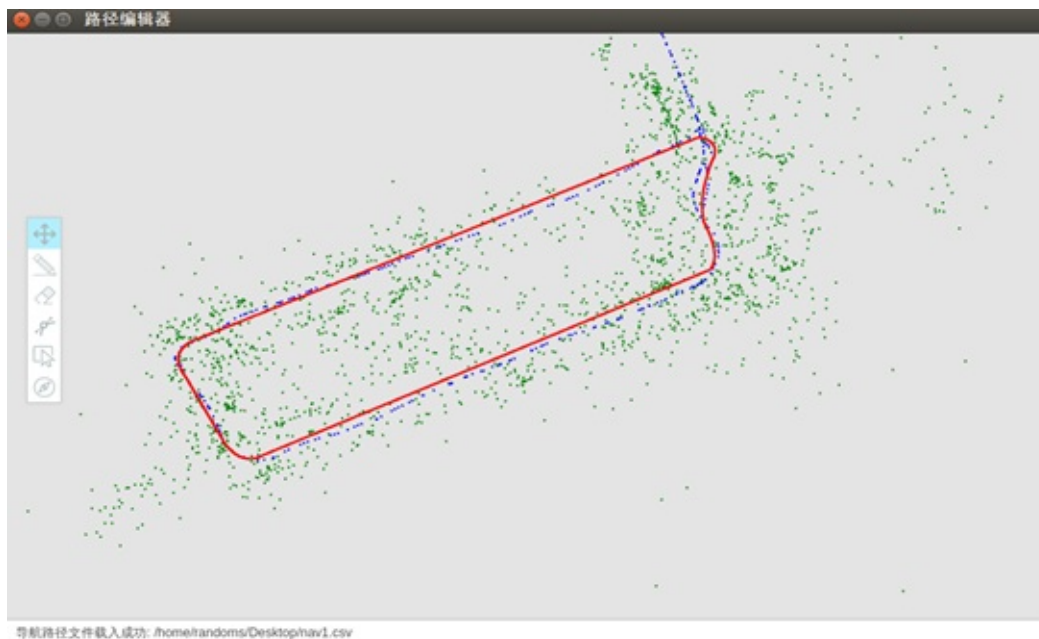


5. Delete tool: if you want to delete a large range of points drawn before, then you can take advantage of this removal tool. Click the Delete tool on the left and click the left mouse button to delete the starting point, you can see in the mouse movement process has a rectangle has been following. Click the left mouse button again to delete the rectangle's selected range. Right-click to cancel the selection.

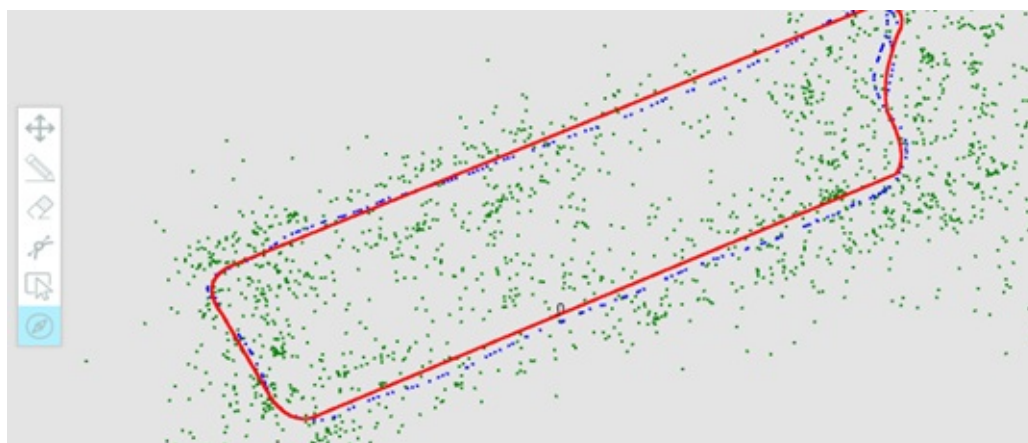
Using these tools, you can draw xiaoqiang's navigation path. Be careful to draw the line as far as possible along the original trajectory, so that the path is unobstructed during the movement. From the green map point you can see the terrain, based on the information to draw the points allowed by the range of motion.

Set navigation key points

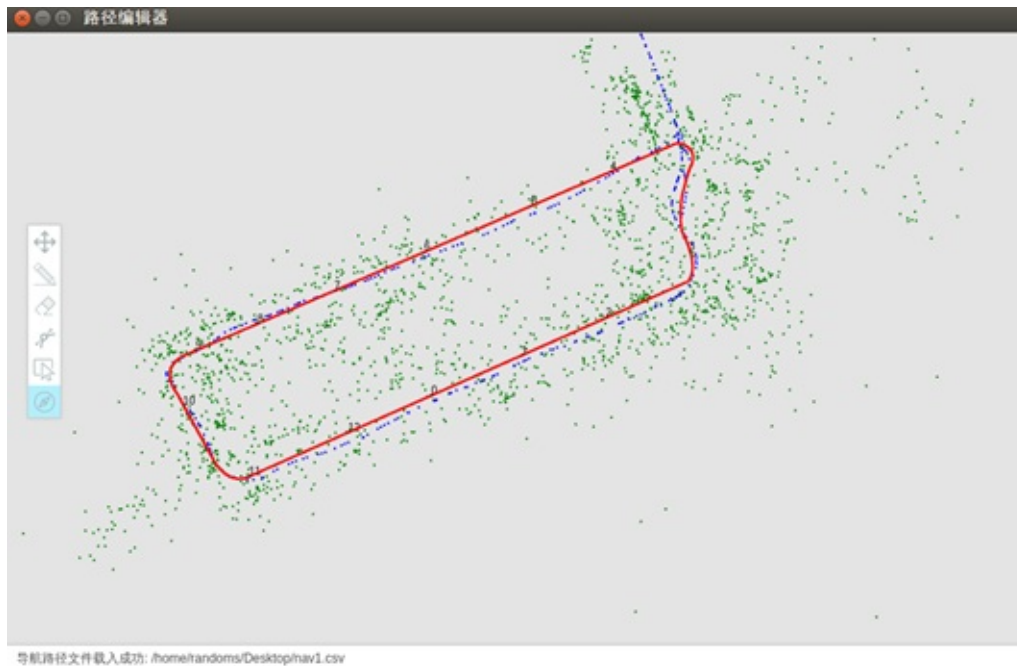
For more complex graphics, there are many ways to move. For example, a figure-eight path, Xiaoqiang may move around one of the circles first, then move around another circle, or two circles may cross each other. Therefore, it is necessary to specify the specific way of the Xiaoqiang movement. Let's take a circular trajectory as an example. In a circular trajectory, Xiaoqiang can move clockwise and also counterclockwise.



Click the navigation point settings button at the bottom of the toolbar on the left. Then start marking key points. Feel free to click on a point on the navigation route and you can see that there is a 0 at this point. This indicates that point 0 has been added here.



If you want to make Xiaoqiang counterclockwise, you can mark a point on the right side. Just add the key points in turn



Click the right mouse button to delete a recently added navigation point. You can also use the eraser and delete tools to delete navigation points. Xiaoqiang will move in the order of the key points.

Export data

1. Export the navigation path file After the navigation path is drawn, click `File -> Export Navigation Path File` in the menu in the upper left corner and select the location where the file is saved. After exporting the file, you can also import it from the menu for secondary editing.
2. Export Navigation Key File After the navigation key is drawn, click `File -> Save Navigation Point` in the menu in the upper left corner and select the file save location. After exporting the file, you can also import it from the menu again for secondary editing. Note: Navigation keys can only be imported after the navigation path file has been successfully imported.

The exported data can be placed in Xiaoqiang's corresponding folder to start visual navigation.

[Xiaoqiang Homepage](#) [Back To Index](#)