



SCOUT MINI R&D Kit Pro

SCOUT MINI Research & Development Kit Pro

# 0 Product Overview



## SCOUT MINI R&D Kit Pro

The SCOUT MINI Research & Development Kit Pro is a ROS development kit designed and developed by Agliex Robotics for entry-level and advanced ROS developers in the field of scientific research and education. It comes equipped with high-performance industrial control, high-precision LiDAR, and multiple sensors based on the Agliex Robot ROS ecosystem. These features enable it to perform various functions such as mobile robot motion control, communication, navigation, and map building, among others.

In addition to its technical capabilities, this kit is also lightweight, portable, and boasts a highly technological industrial design. It even includes an exclusive sensor customized holder. With complete developer documentation and DEMO resources, it serves as the ideal experimental platform for rapid secondary development of ROS for multi-directional applications like education and scientific research, product pre-research, subjects, and product demonstration.

# 1 Configuration List and Parameters

## 1.1 Shipping List

### 1.1.1 SCOUT MINI Pro

Name	Quantity	Model
IPC	1	Nvidia AGX Xavier
Laser Sensor	1	RS-Helios-16P
Vision Sensor	1	Intel RealSense D435
Router	1	GL.iNet GL-A1300
Chassis	1	SCOUT MINI(off-road)
Remote Controller	1	FS-I6S
Charger	1	Agilex UY360
Voltage Stabilizer	1	24V To 12V
Voltage Stabilizer	1	12V To 5V
HD Display	1	
USB To CAN	1	
USB HUB	1	



RS-Helios-16P



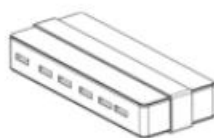
Nvidia AGX Xavier



Inter RealSense D435



Router



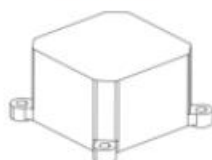
USB-HUB



USB To CAN



SCOUT MINI



IMU(Optional)



Display

SCOUT MINI Pro Main configuration

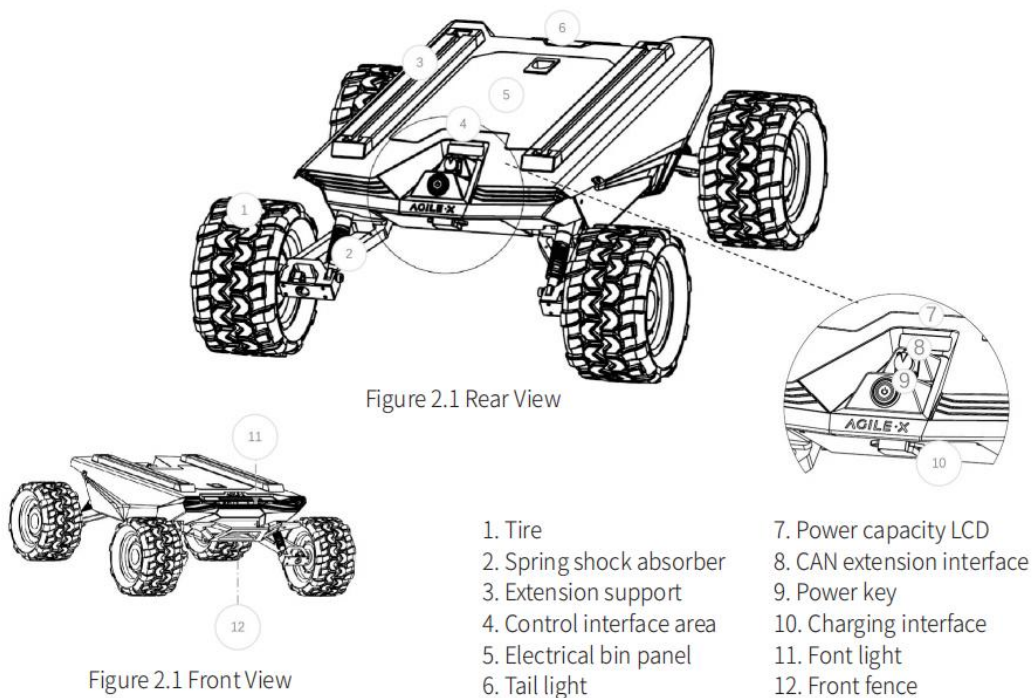
## 1.2 SCOUT MINI Introduction

### 1.2.1 SCOUT MINI

The SCOUT MINI mobile chassis uses 4WD design with powerful off-road performance, compact design and truly 'smart like a swallow, galloping like a heart'. This innovative design inherits the advantages of the 4WD differential chassis series from SCOUT, which features independent suspension and zero turning radius. The hub motor has also been improved upon in this model.

With a minimum turning radius of 0m and a climbing angle almost reaching 30 degrees, SCOUT MINI performs exceptionally well off-road despite being 50% smaller than the original SCOUT. Its power control system offers precise, stable, and controllable power, allowing it to achieve high speeds of up to 10.8km/h.

The SCOUT MINI development platform comes equipped with its own control system that supports standard CAN protocol and can connect to various external devices. It also supports ROS/Autoware secondary development and advanced robotics development. Standard accessories include an activation plug, 24V@15Ah lithium battery, and an endurance mileage of up to 10KM.



## 1.2.2 Get started quickly with Scout Mini

### 1.2.2.1 SCOUT MINI Checking

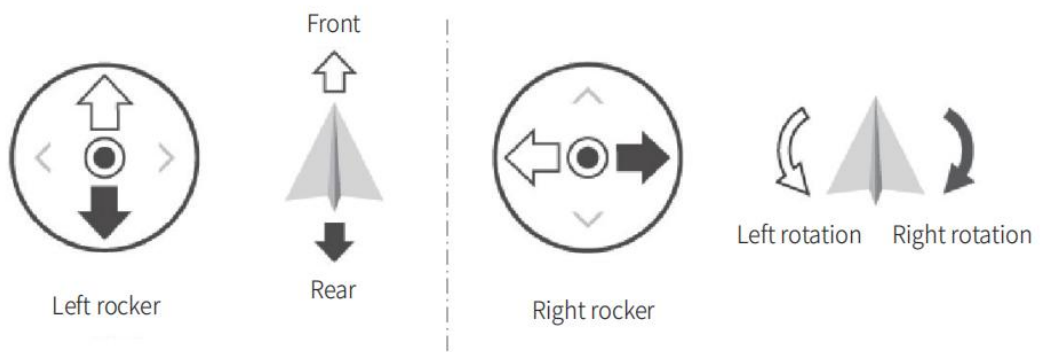
- Press power button and wait for a few seconds
- Check the display to see if the power is lower than 30%. If it is lower than 30%, please charge it.
- If the battery level is less than 30%, please use the standard charger to charge the vehicle. Turn off the power of the vehicle during charging.
- It takes about 1 and a half hours to charge from low battery to full power.

### 1.2.2.2 Remote Control

- Check whether the battery of remote controller has been installed.
- Move the joysticks SWA, SWB, SWC, and SWD to the top.
- Press and hold the power switch buttons 1 and 2 at the same time until the power is turned on.

### 1.2.2.3 Remote Control Operation

The remote control has preset switches default setting. Please do not change the switches setting. Any changing may cause control failure. SWB switch the control mode, the SWC is controlling the light on and off while SWD controls the speed mode. The left joystick controls the Scout mini move forward and backward while the right joystick controls the rotation. Please note that the internal mapping motion of the chassis is mapped based on percentage, so when the joysticks are at the same position, the speed is constant.



- SWB is the control mode switching button. The remote controller joystick is at the top for command mode, at the middle for remote control mode, and at the bottom for constant speed mode.
- SWC is the light controlling button. When it is at the bottom position, it is closed, when it is in the middle, the light is open, and the top position is the breathing light

mode. Please note that the lighting control option setting is only valid under the remote controller mode.

- SWD is the speed gear selection mode, the up position is the low gear mode (the fastest speed is about 10km/h), and bottom position is the high gear (the fastest speed is about 20km/h). Please note that the gear setting button is only valid under the remote control mode.

#### **1.2.2.4 SCOUT MINI Preparation**

- Please use SCOUT MINI in a relatively open area for the first time to avoid any inappropriate operation and damage.
- Press the SCOUT MINI power button and wait a few seconds.
- Set SWB to the middle position.
- Try to switch the light mode manually to make sure that the mode is selected correctly.
- Try to gently push the left joystick forward, you can see that the car is moving forward slowly.
- Try to gently push the left joystick back, you can see that the car is moving backwards slowly.
- Release the left joystick and the vehicle will stop.
- Try to gently push the right joystick to the left, you can see that the car slowly rotates to the left.
- Try to gently push the right joystick to the right, you can see that the car slowly rotates to the right.
- Release the right joystick and the vehicle will stop;
- Then try to control the vehicle in the relatively open area and be familiar with the speed control of the vehicle.

#### **1.2.2.5 Turn off SCOUT MINI**

- Press SCOUT MINI power button and then release.

#### **1.2.2.6 Turn off Remote Controller**

- Hold the power button 1 and 2 at the same time for few seconds until turned off.

### 1.2.3 SCOUT MINI Parameters

SCOUT MINI Parameters Description	
Size	627mm × 550mm × 252mm
Wheelbase	452mm
Track width	450mm
Weight	20kg
Minimum Ground Clearance	107mm
Standard Load	10kg(Tested on the ground, the friction coefficient is 0.5)
Maximum Speed	20km/h
Minimum Turning Radius	0(Can rotate in place)
Maximum climbing angle	30°(With load)
Obstacle avoiding	70mm
Maximum mileage	10km(Without load)
Driver mode	4WD
Temperature	-20°C~60°C
Charger	AC 220
Charging time	1.5H
Voltage Output	24V
Battery	24V/15Ah
Code Wheel	1024 Photoelectric Incremental
Communication interface	Standard CAN
IP protection	IP22(IP64 for customization)
Suspension	Independent swing arm suspension

## 1.3 Nvidia Xavier Introduction

NVIDIA® Jetson AGX Xavier™ sets a new standard for compute density, energy efficiency, and AI inference capabilities in edge devices. This cutting-edge technology represents the next evolution of intelligent machines with end-to-end autonomous capabilities. Despite being only 100 x 87 mm, the Jetson AGX Xavier delivers the same performance as a larger workstation while being one-tenth the size. This makes it the perfect solution for

autonomous machines such as delivery and logistics robots, factory systems, and large industrial UAVs.

As the first computer designed specifically for autonomous machines, Jetson AGX Xavier is powerful enough to perform complex tasks such as visual odometry, sensor fusion, localization and mapping, obstacle detection, and route planning algorithms that are crucial for next-generation robots. It offers GPU workstation-class performance in a compact form factor, featuring 32 teraflops of peak computing power and 750 Gbps of high-speed I/O performance.

Jetson AGX Xavier offers new levels of compute density, energy efficiency, and AI inference capabilities at the edge. Users can configure their applications for 10-watt, 15-watt, or 30-watt operating modes using the Jetson AGX Xavier module.

Developer Kit Technical Specifications	
GPU	Tensor Core 512 -Volta GPU
CPU	8-core ARM v8.2 64-bit CPU, 8 MB L2 + 4 MB L3
Memory	32 GB 256-bit LPDDR4x   137 GB/sec
Storage	32 GB eMMC 5.1
PCIe X16	X8 PCIe Gen4/x8 SLVS-EC
RJ45	Gigabit Ethernet
USB-C	2 USB 3.1 interfaces, DP interface (optional), PD interface (optional). Supports closed system debugging and programming through the same port.
Camera Interface	(16 *) CSI-2 channels
M.2 Key M	NVMe
M.2 Key E	PCIe x1 + USB 2.0 + UART (for Wi-Fi/LTE) / I2S + DMIC + GPIOs
40 Pin Header	UART + SPI + CAN + I2 C + I2 S + DMIC + GPIOs
HD Audio connector	HD Audio
eSTATp + USB 3.0 Type A	SATA interface with PCIe x1 bridge + USB 3.0 (Data for 2.5-inch SATA interface)
HDMI Type A	HDMI 2.0
μSD/UFS	SD/UFS





## 1.4 Intel RealSense D435

Binocular vision sensors play a critical role in various applications within the field of robot visual measurement, visual navigation, and other areas of the robotics industry. At present, we have carefully chosen visual sensors that are widely used in scientific research and education. Among them, the Intel RealSense Depth Camera D435 stands out with its global image shutter and expansive field of view, enabling efficient capture and streaming of depth data for moving objects. This camera delivers exceptionally precise depth perception, making it an excellent choice for mobile prototypes.

	Item	Intel Realsense D435
Features	Scenes	Indoor/Outdoor
	Measure Distance	About 10 m
	Type of Depth Shutter	Global Shutter, 3um X 3um pixel size
	IMU Support	No
	Depth Camera	Active Infrared
Depth Camera	FOV	86° x 57° (±3°)
	Depth Frame Rate	0.105m
	Depth Resolution	1280 x 720
	Maximum Measurement Distance	About 10 m

RGB	Depth Frame Rate	90 fps
	Resolution	1280 x 800
	FOV	69.4° × 42.5° (±3°)
	Frame	30fps
Others	Size	90mm x 25mm x 25mm
	Interface Type	USB-C 3.1

## 1.5 IMU (Optional)

The CH10X series is an attitude-sensing system that uses high-performance, small-volume MEMS inertial devices to sense object attitude information. It integrates an inertial measurement unit (IMU), a magnetometer, and a microcontroller equipped with an extended Kalman fusion algorithm (EKF) device. It can output three-dimensional orientation data based on local geographical coordinates calculated by the sensor fusion algorithm, including absolute reference heading angle, pitch angle, and roll angle. Calibrated raw sensor data can also be output. The IP68 waterproof shell-type package can be easily integrated into the user's system.

### Attitude angle output accuracy

Attitude angle	Typical Value
Roll angle/pitch angle-static error	0.8°
Roll angle/pitch angle-dynamic error	2.8°
Heading and angle accuracy during movement	3°

### Gyroscope

Parameters	Value
Measuring range	±2000°/s
Bias stability	10°/h
Scale non-linearity	±0.1% (full range)
Noise density	0.08
Acceleration sensitivity	0.001°/s/g

### Accelerometer

Parameters	Value
Measuring range	±8G (1G=1x gravity acceleration)
Bias stability	30mG
Non-linearity	±0.5% (full range)
Noise density	120

### Magnetic sensor parameters

Parameters	Value
Measuring range	±8G (Gauss)
Non-linearity	±0.1%
Resolution	0.25mG



## 1.6 RS-Helios-16P

RS-Helios-16P is a 16-line lidar mass-produced by RoboSense. It is the first domestic and world-leading small lidar. It is mainly used for autonomous driving vehicle environment perception, and robot environment perception, human-machine surveying, and mapping. RS-Helios-16P has 16 built-in laser components. It emits and receives high-frequency laser beams at the same time. Through 360° rotation, it performs real-time 3D imaging and provides accurate three-dimensional space point cloud data and object reflectivity, allowing the machine to obtain reliable Environmental information and provide a strong guarantee for positioning, navigation, obstacle avoidance, etc.

Item	Parameters
Range	0.2m~150m(110m@10% NIST)
Range Accuracy	±2cm
Points Per Second	288,000pts/s(Single Return) 576,000pts/s(Dual Return)
Field of View (Vertical)	-15°~ + 15°
Vertical Resolution	2°
Frame Rate	10Hz/ 20 Hz

Laser Safety Classification	Class 1
Weight(Without cabling)	~1.0kg
Power Consumption	11W
Voltage	9V~32V
Operating Temperature	-30°C ~ + 60°C



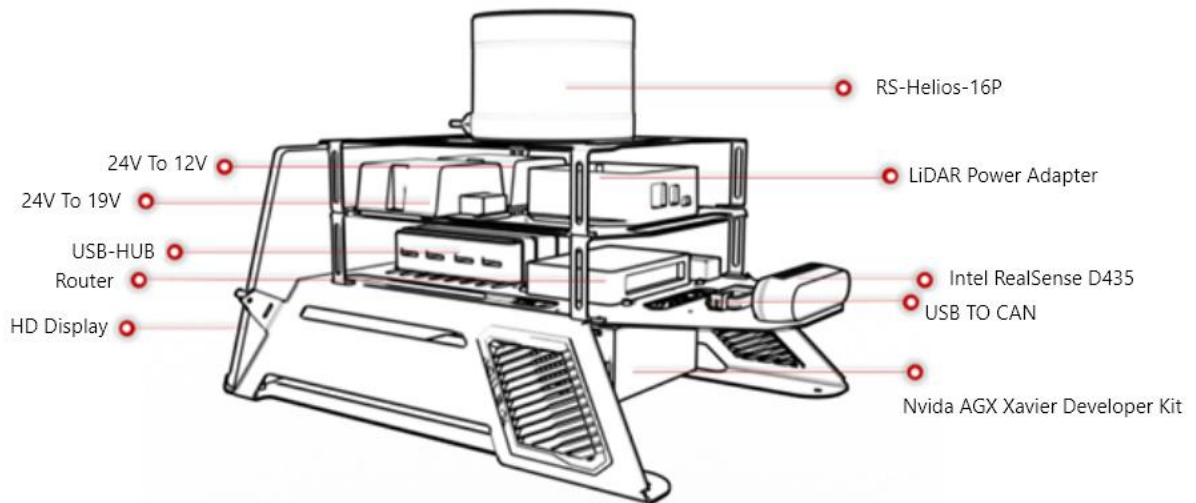
## 2 Instructions for Hardware Installation and Electricity

### 2.1 SCOUT MINI Pro Kit Installation

#### 2.1.2 SCOUT MINI Pro

The SCOUT MINI Pro scientific research and education kit bracket features a stacked design, providing a well-organized structure. The frame utilizes sheet metal for support, allowing customers to easily expand connections. A high-definition display is conveniently positioned at the rear for customer debugging purposes. The bottom section incorporates a hollow sheet metal bracket, which is assembled with the standard

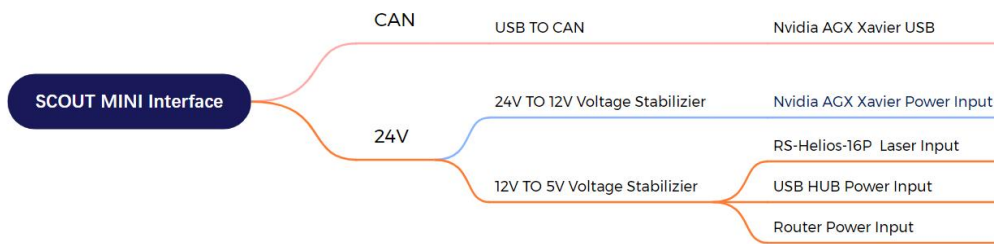
aluminum profile bracket on top of the SCOUT MINI. The computing unit of the SCOUT MINI Pro is located on the first floor. The second layer primarily consists of the USB-HUB expansion interface, USB to CAN module, and Intel RealSense D435. The third layer houses the voltage stabilizing module, while the top layer currently accommodates a VLP 16 lidar. For specific installation locations and schematic diagrams, please refer to the figure provided below.



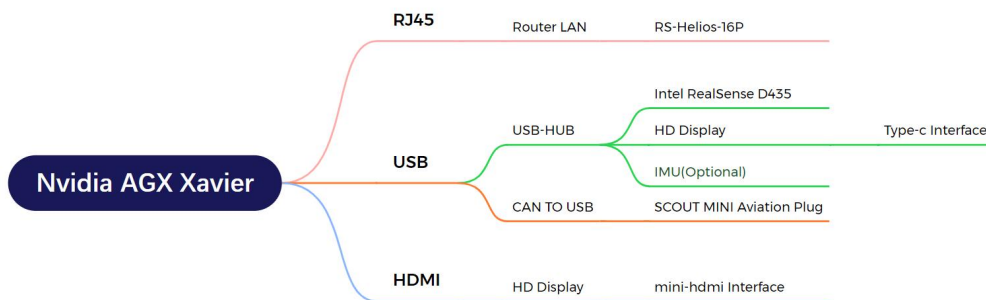
## 2.2 Electricity and Communication Connection

### 2.2.2 SCOUT MINI Pro

SCOUT MINI Pro provides power and communication interfaces for upper equipment through the aviation expansion interface of the chassis. The power supply of the SCOUT MINI chassis power expansion interface (maximum power output supports 24V@5A) comes from the battery of the chassis. Without voltage stabilization and voltage regulation modules, in order to power other expansion equipment, it is necessary to add voltage stabilization and voltage regulation equipment. To facilitate later use, we chose 19V and 12 V voltage stabilizing modules. The 19V voltage stabilizing module mainly provides power input for Nvidia AGX Xavier, and the 12V voltage stabilizing module provides power input for VLP-16 lidar, USB HUB, and wireless routers.



The external topological connection of Nvidia AGX Xavier, serving as the core computing unit, is as follows. The network port of Xavier is connected to the router's network port, enabling remote desktop connection, access, and debugging. This setup also allows for the expansion of other network devices. Additionally, an externally added USB-HUB with independent external power supply is incorporated. The USB-HUB facilitates the connection of D435 binocular camera, LCD screen, and other peripherals.



## 2.3 Sensor Expansion

External expansion mainly involves mechanical installation, power supply, and communication expansion. In terms of power supply expansion, we carefully considered this issue during the early stage of selecting the power supply voltage stabilizing module. As a result, the module has a certain margin reserved to accommodate potential power needs. For communication expansion, we added a USB-HUB and wireless gateway to the device to enable access to more devices.

# 3 Development Guide

## 3.1 ROS Development Introduction

### 3.1.1 ROS History

The rapid advancement of hardware technology has not only facilitated the progress and

complexity of the robotics field but also presented significant challenges to the software development of robotic systems. With an increasing abundance of robot platforms and hardware devices, there is a growing demand for software code reusability and modularity. However, existing robot systems often fail to meet these requirements adequately. In comparison to hardware development, software development in this field has been comparatively insufficient.

To address the significant challenges faced by robot software development, developers and research institutions worldwide have devoted resources to the creation of universal software frameworks for robots. In recent years, several remarkable robot software frameworks have emerged, greatly facilitating software development efforts. Among these frameworks, the Robot Operating System (ROS) stands out as one of the most exceptional.

ROS is a flexible framework designed for the creation of robot software. It seamlessly integrates numerous tools, libraries, and protocols, offering functionalities akin to those provided by operating systems. These functionalities include hardware abstraction description, management of underlying drivers, execution of shared functions, inter-program message passing, and program release package management. ROS effectively simplifies the creation of complex tasks and enables stable behavior control across diverse and intricate robot platforms.

### **3.1.2 ROS Concept**

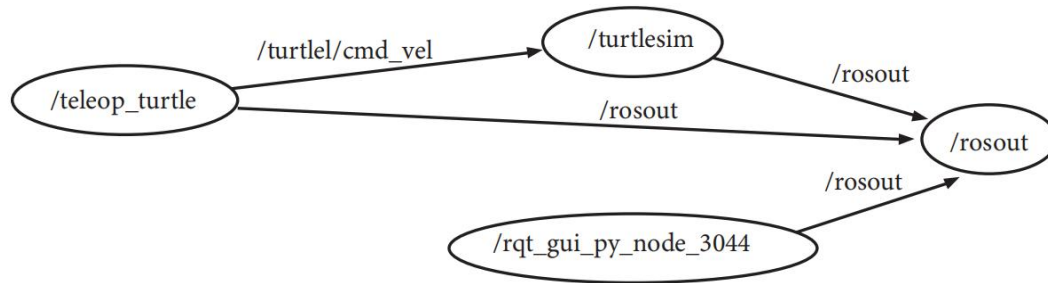
The concept of ROS includes three layers: the file system layer, computing layer, and community layer. The primary objective of ROS is to enhance software reusability in robot research and development. To achieve this goal, ROS adopts a distributed structure, enabling each functional module within the framework to be designed, compiled, and executed independently, while being loosely coupled with other modules.

ROS primarily offers functionalities and mechanisms such as hardware abstraction, underlying drivers, message passing, program management, and application prototypes for robot development. It also integrates various third-party tools and library files to assist users in swiftly establishing, writing, and integrating robot applications across multiple machines. Moreover, the functional modules within ROS are encapsulated within independent function packages (Packages) or meta-function packages (Meta Packages), facilitating sharing and distribution within the community.

### **3.1.3 ROS Node**

A node represents a single process running the ROS graph. Every node has a name, which it registers with the ROS master before it can take any other actions. Multiple nodes with different names can exist under different namespaces, or a node can be defined as anonymous, in which case it will randomly generate an additional identifier to add to its

given name. Nodes are at the center of ROS programming, as most ROS client code is in the form of a ROS node which takes actions based on information received from other nodes, sends information to other nodes, or sends and receives requests for actions to and from other nodes.

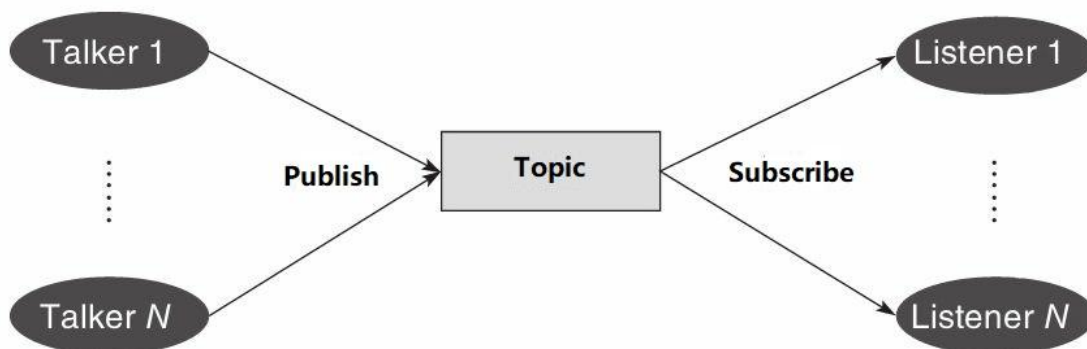


### 3.1.4 ROS Message

The most important communication mechanism between nodes is message communication based on the publish/subscribe model. Each message is a strict data structure, supporting standard data types (integer, floating point, Boolean, etc.), and also supports nested structures and arrays (similar to the structure struct in C language). It can also be configured based on Requirements are defined by developers themselves.

#### 3.1.4.1 ROS Topic

Messages are delivered in a publish/subscribe manner. A node can publish messages for a given topic (called a publisher/Talker), or it can pay attention to a topic and subscribe to specific types of data (called a subscriber/Listener). Publishers and subscribers do not know each other's existence. There may be multiple nodes in the system publishing or subscribing to messages on the same topic at the same time.



#### 3.1.4.2 ROS Service

Although the topic-based publish/subscribe model is a very flexible communication model, it is not suitable for the two-way synchronous transmission model. In ROS, we call this



synchronous transmission mode Service, which is based on the Client/Server model and contains two parts of communication data types: one for requests and the other for responses. Similar to a web server. Unlike topics, only one node in ROS is allowed to provide a specified named service.

### **3.1.4.3 ROS File System**

Similar to an operating system, ROS organizes all files according to certain rules, and files with different functions are placed in different folders.

Function package (Package): Function package is the basic unit in ROS software, including ROS nodes, libraries, configuration files, etc.

Package Manifest: Each function package contains a package manifest named package.xml, which is used to record the basic information of the function package, including author information, license information, dependency options, compilation flags, etc.

Meta Package: In the new version of ROS, the concept of the original function package set (Stack) is upgraded to a "meta function package", whose main function is to organize multiple function packages for the same purpose. For example, a ROS navigation meta-function package will include multiple functional packages such as modeling, positioning, and navigation.

Meta-function package list: Similar to the function package list, the difference is that the meta-function package list may include function packages that need to be depended on at runtime or declare some reference tags.

## **3.2 Start up and shut down**

### **3.2.1 Installation**

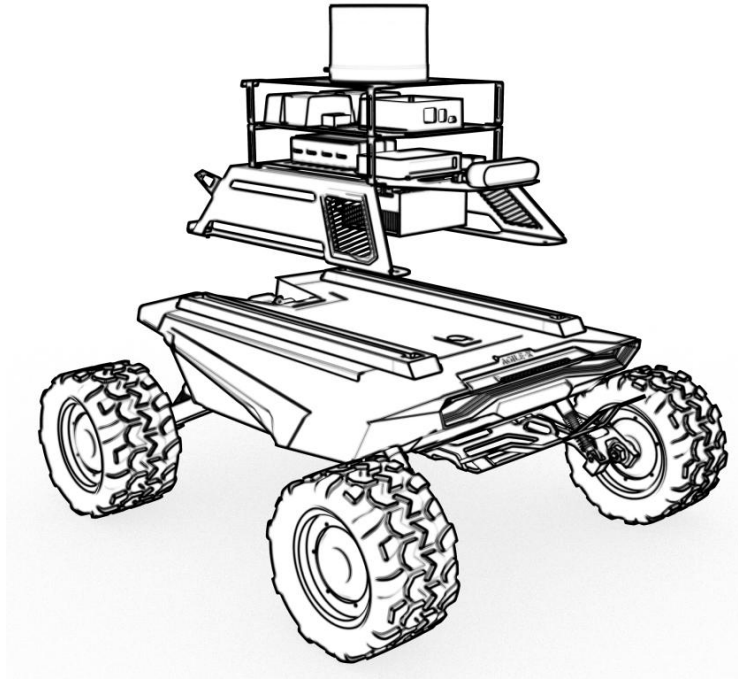
#### **3.2.1.1 Tool**

Set of hexagonal tools.

#### **3.2.1.2 Sensors holder and mobile base**

##### **SCOUT MINI Pro**

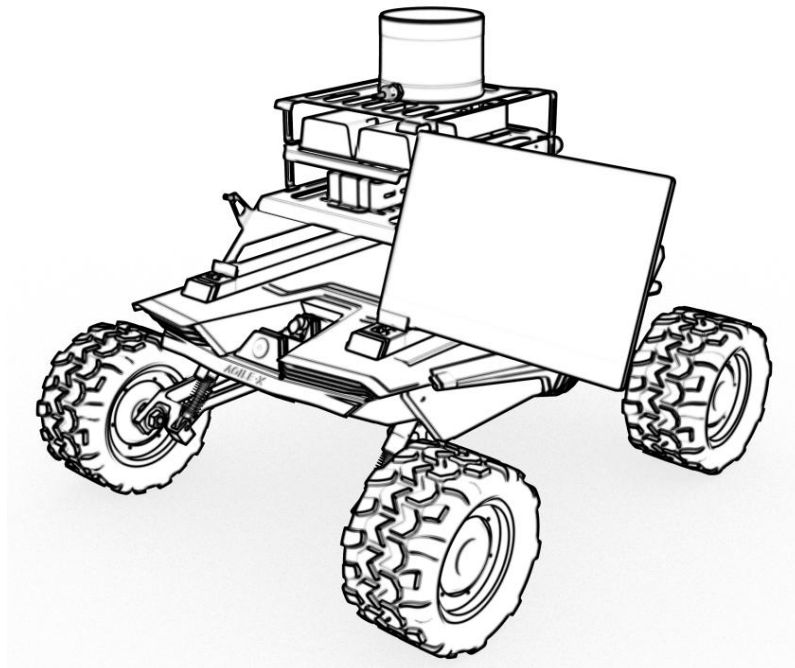
When the product is ready for shipping, the sensor holder is separated from the Scout MINI. User need to use tool to set the sensor holder on the Scout MINI. Firstly, put the four slider nuts into the slide way each side on the Scout MINI , and then use the hexagonal tool to screw corresponding four screws on the holder to the platform. Please refer to the figure shown below.



### 3.2.1.3 HD Display Installation

#### SCOUT MINI Pro

Put the HD display into the display holder horizontally, as shown in the figure below. Then settle the display holder with screws and nylon posts to the holes on both sides.



## 3.2.2 Checking before start up

- Check whether any wiring harness connections are disconnected;
- Please make sure to operate in a relatively open area without large area of water . The environment is relatively open and stable, there are no flammable, explosive and other dangerous goods around;
- The whole machine is complete, the wiring harness is intact and there is no break, and the sensors are not damaged.

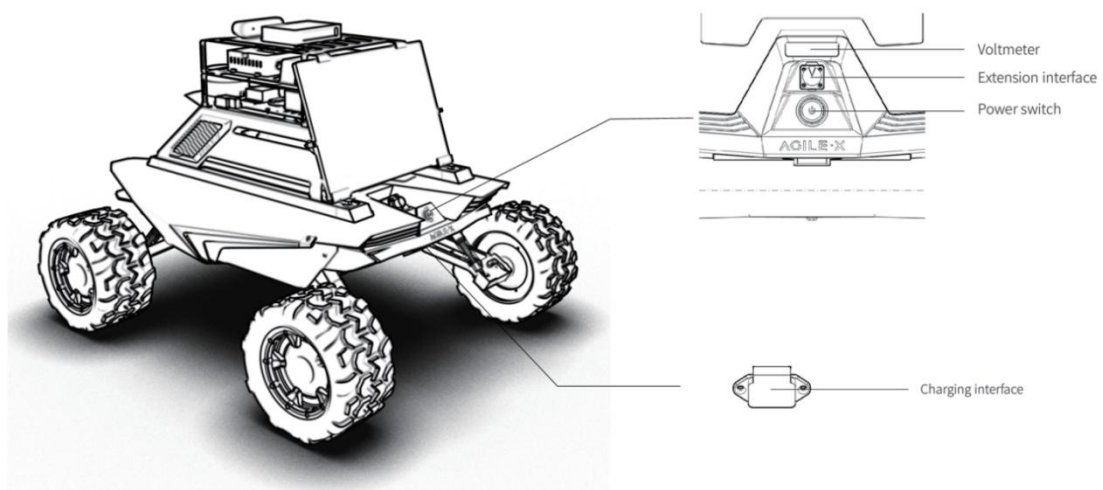
## 3.2.3 Mouse and keyboard

This Kit is not provided with mouse and keyboard, users can purchase it based on requirement, and can be connected with the USB interface of the computing unit or the USB HUB.

## 3.2.4 Power on

### 3.2.4.2 SCOUT MINI Pro

Press the power button on the Scout MINI, as shown in the figure below.



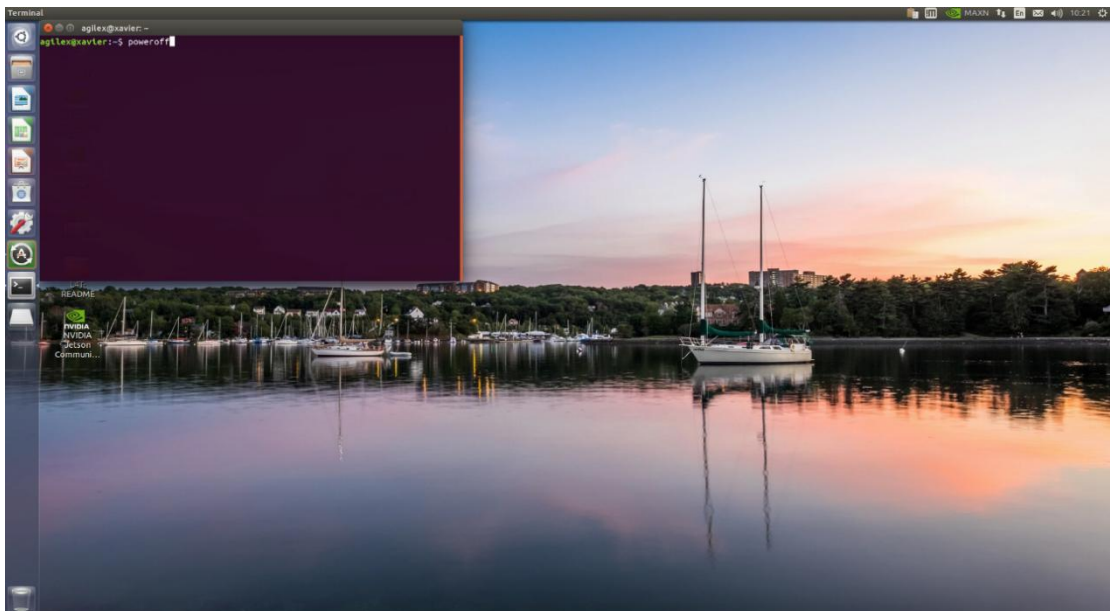
## 3.2.5 Computing Unit Login

After pressing the power button of the Scout MINI , the computing unit will automatically log in and show the following interface, the root authority password and system login password are agx .



### 3.2.6 Shut Down

If you need to shut down the system, do not press the power button directly because the computing unit is running. Enter the command `$poweroff` in the terminal or click shut down of the drop-down menu, as shown in the figure below, wait until the display shows no signal input, then press the power button to shut down the system.





### 3.3 SCOUT MINI Pro Development Environment

(Ubuntu, ROS, Gazebo, RVIZ)

Ubuntu	18.04
ROS	melodic
Gazebo	9.0
RVIZ	-

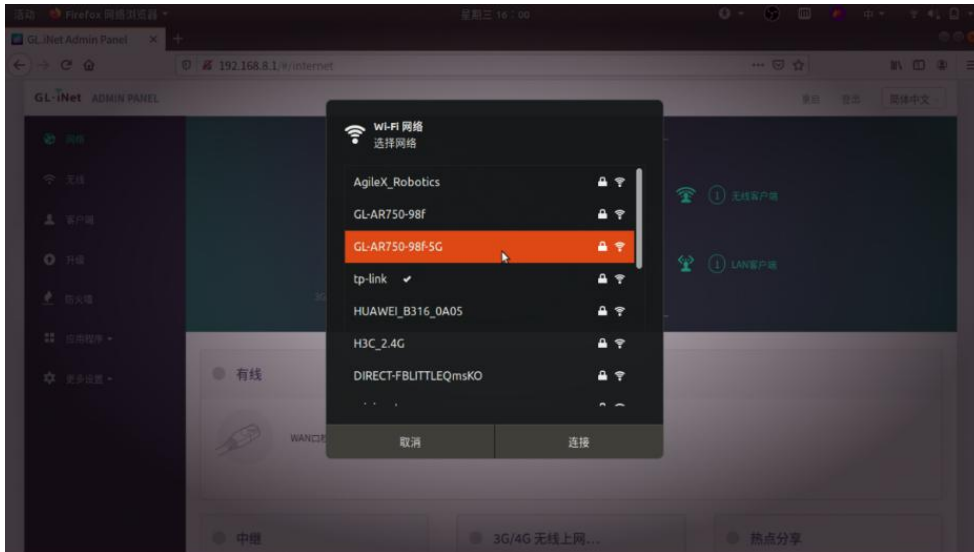
It has passed the test and can be used normally under the above environment.

### 3.4 Remote Desktop

#### 3.4.1 Download the corresponding operating system remote

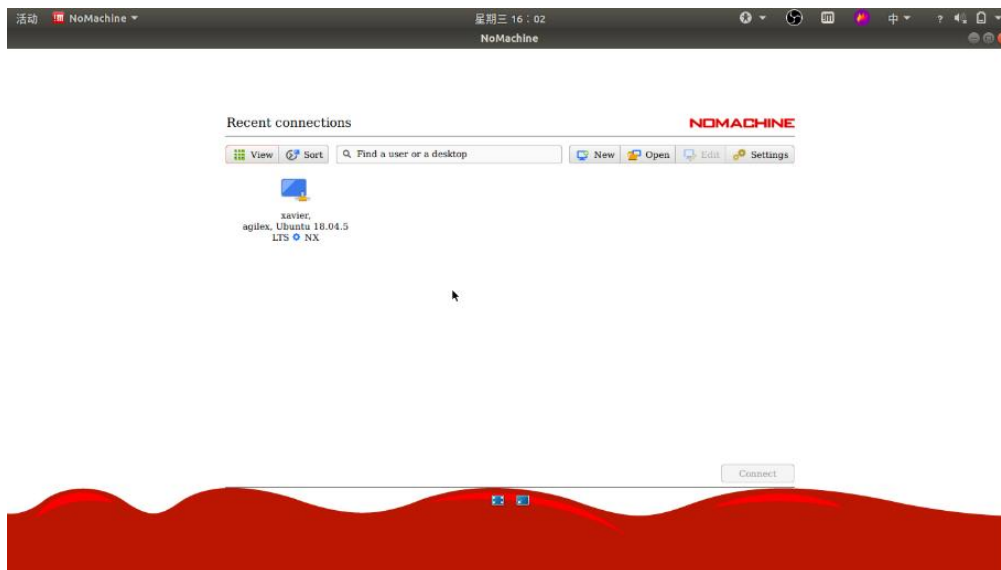
#### desktop software on the host computer

Open the wireless network settings of the host computer, it shows that there are two signals from the router GL-AR750(respectively), please connect to the 5G frequency band, the default password is *goodlife*

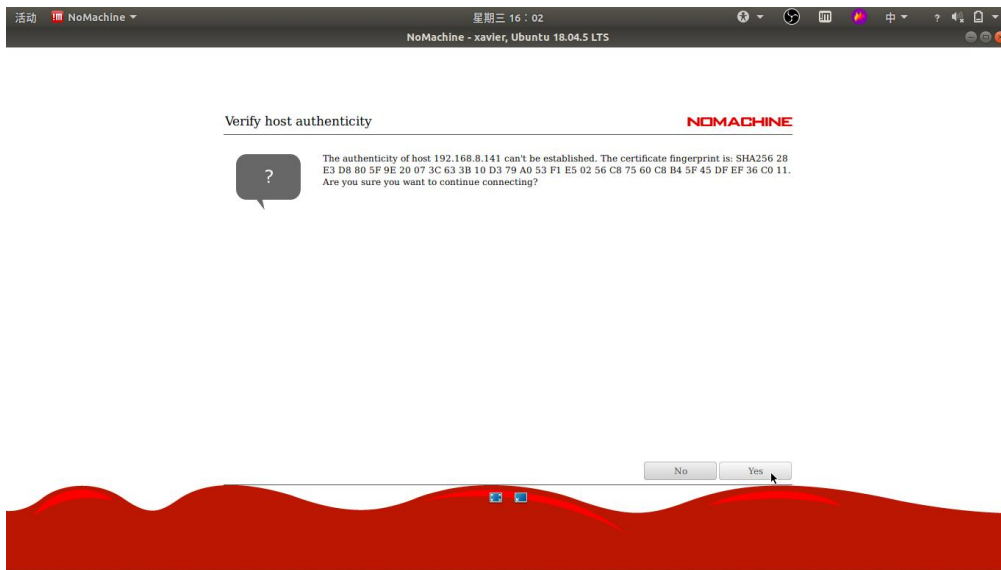


### 3.4.2 Xavier connection (make sure the computer unit is power on)

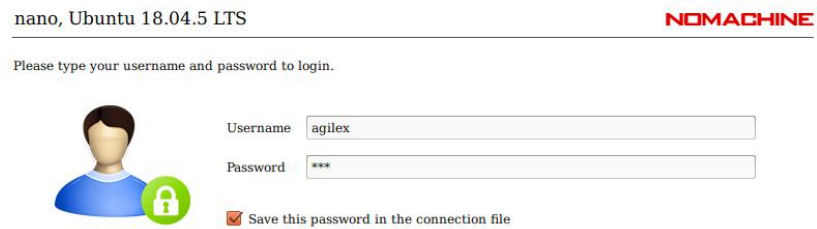
#### 3.4.2.1 Choose connection object



### 3.4.2.2 Click Yes



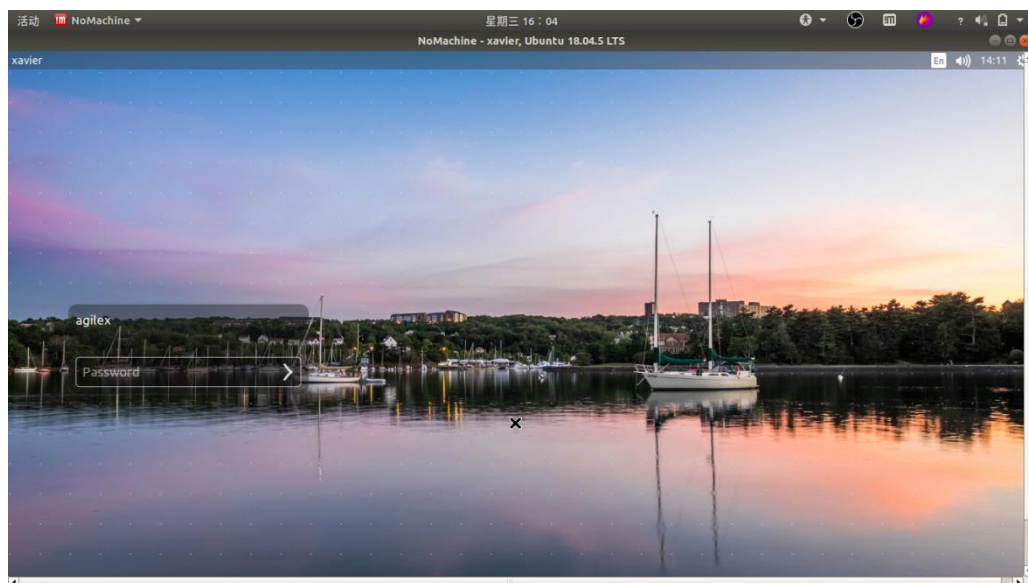
### 3.4.2.3 Username: agilex Password: agx Choose to save the password.



### 3.4.2.4 Keep clicking the default OK

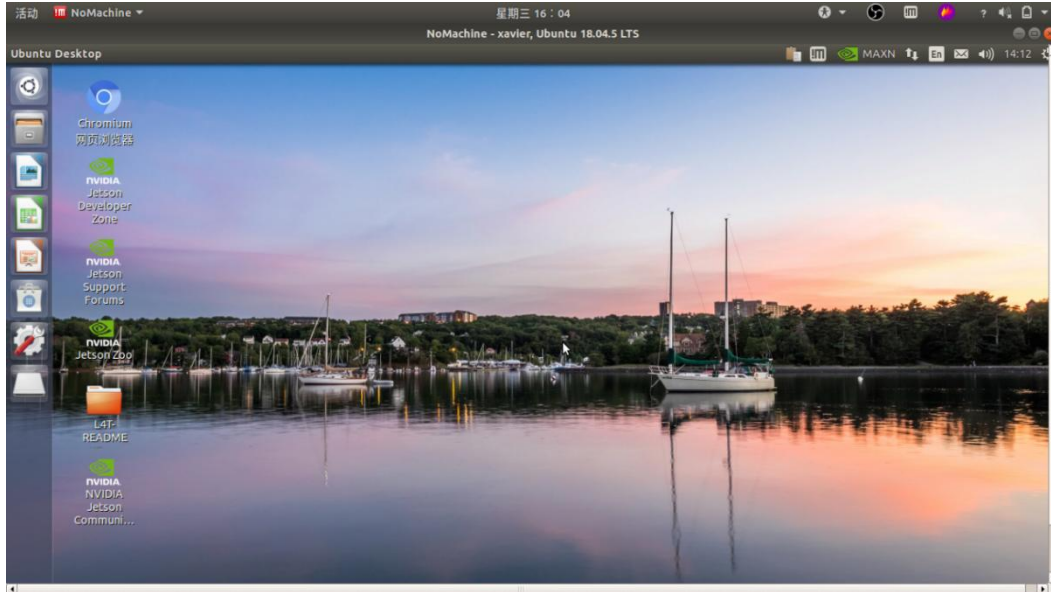


### 3.4.2.5 Enter the default password agx





### 3.4.2.6 Successful connection



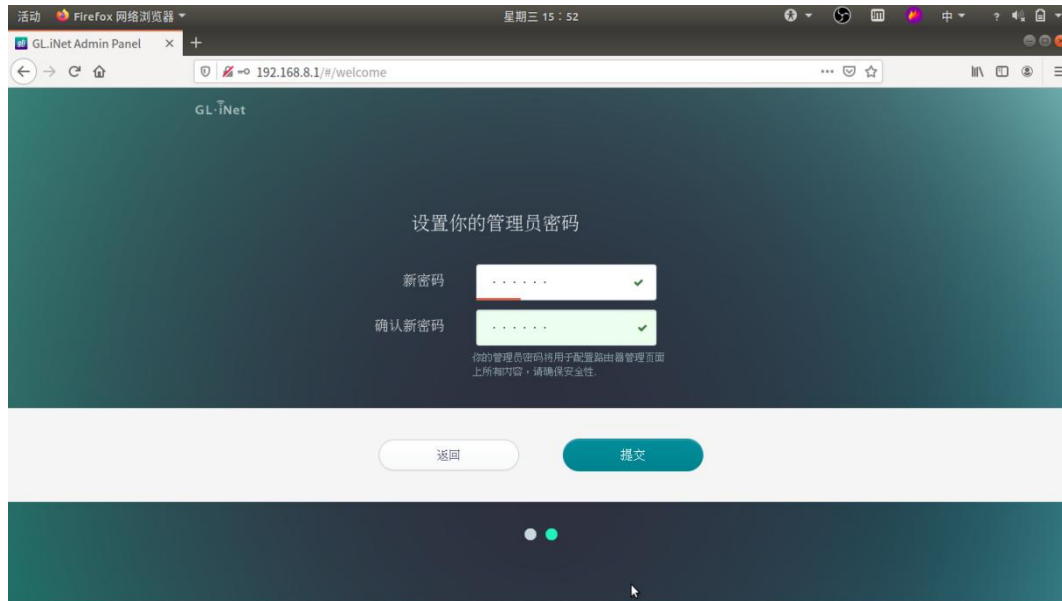
### 3.4.3 Network port to connect to remote desktop

#### 3.4.3.1 Enter the router, the default address is 192.168.8.1. Select the language and set the password

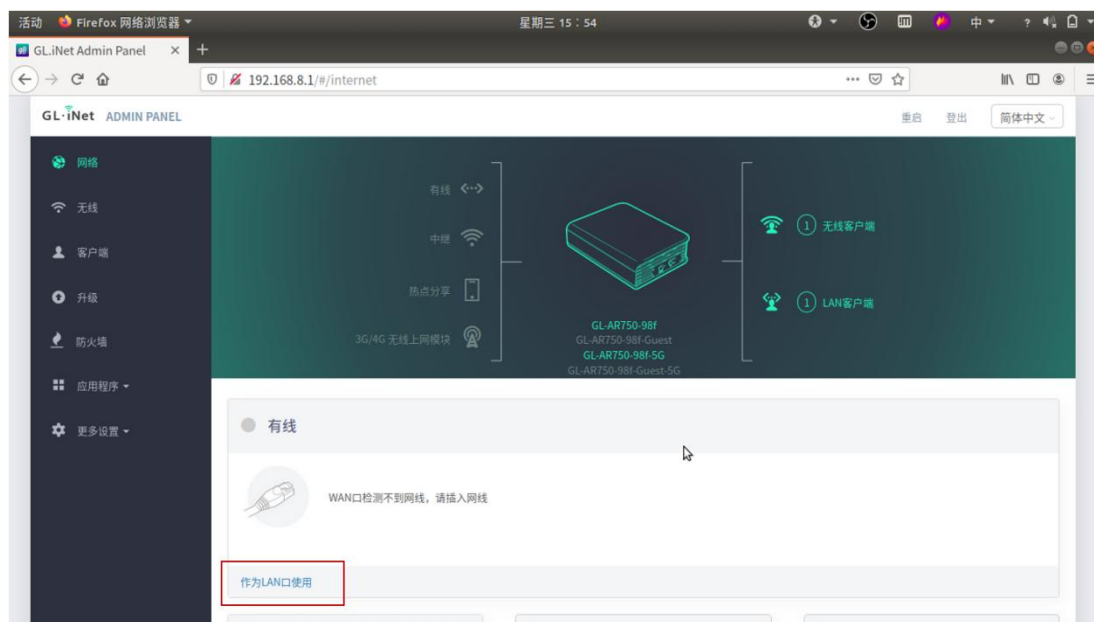
Note: After resetting the router, the default IP address is 192.168.8.1.

Enter, 192.168.8.1 in the browser, select the language, and set the password.

After the test platform is produced, the router password is unified: 12345678, and the IP address is 192.168.1.1



### 3.4.3.2 Click USE as LAN port



3.4.3.3 Successfully set to LAN port mode, now you can connect the Kit with the host computer through a network cable for access



## 3.5 ROS Installation

### Install Source

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

### or source from China

```
sudo sh -c './etc/lsb-release && echo "deb http://mirrors.ustc.edu.cn/ros/ubuntu/$DISTRIB_CODENAME main" > /etc/apt/sources.list.d/ros-latest.list'
```

### Set key

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6B ADE8868B172B4F42ED6FBAB17C654
```

### Update

```
sudo apt-get update
```

### ROS Desktop-Full Installation

#### Ubuntu16.04

```
sudo apt-get install ros-kinetic-desktop-full
```

Ubuntu18.04

```
sudo apt-get install ros-melodic-desktop-full
```

Resolve dependencies

```
sudo rosdep in  
rosdep update
```

Environment Setup

Ubuntu16.04

```
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc  
source ~/.bashrc
```

Ubuntu18.04

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc  
source ~/.bashrc
```

Install rosinstall, a convenient tool.

```
sudo apt install python-roinstall python-roinstall-generator python-wstool build-essential
```

## 3.6 Sensors base node

### 3.6.1 USB To CAN Drive installation and testing

Enable gs\_usb.

```
sudo modprobe gs_usb
```

Open CAN port and set the baud rate.

```
sudo ip link set can0 up type can bitrate 500000
```

If no errors occurred in the previous steps, you should now be able to view the CAN by using command.

```
ifconfig -a
```

Testing the hardware by using can-utils.

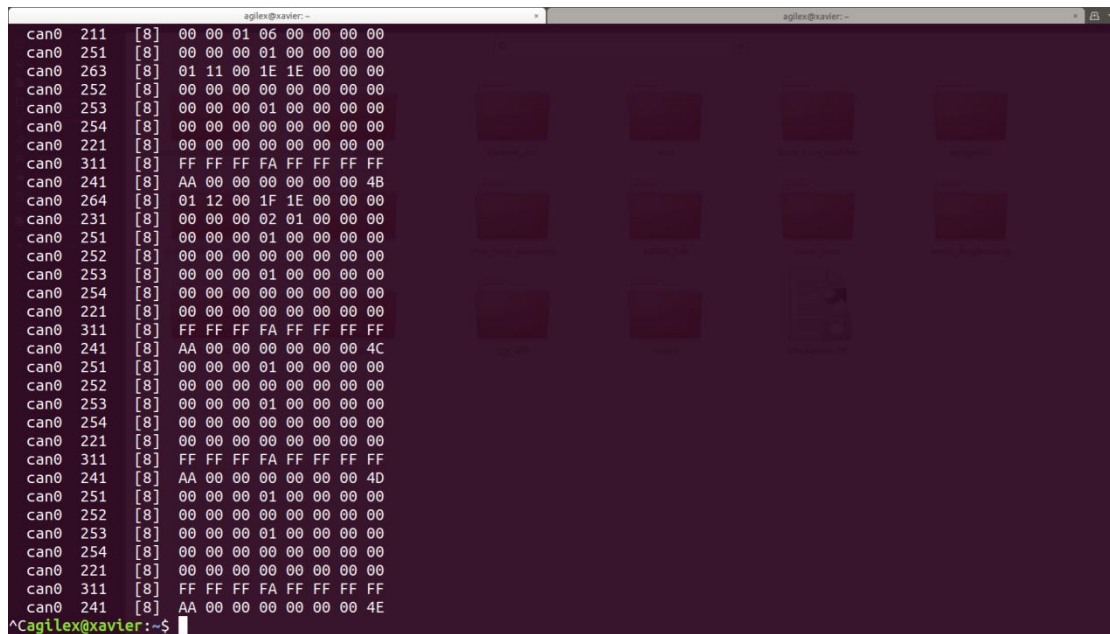
```
sudo apt install can-utils
```

Or run the script:

```
roslaunch scout_bringup setup_can2usb.bash
```

Testing the commands.

```
$ candump can0
```



```
can0 211 [8] 00 00 01 06 00 00 00 00
can0 251 [8] 00 00 00 01 00 00 00 00
can0 263 [8] 01 11 00 1E 1E 00 00 00
can0 252 [8] 00 00 00 00 00 00 00 00
can0 253 [8] 00 00 00 01 00 00 00 00
can0 254 [8] 00 00 00 00 00 00 00 00
can0 221 [8] 00 00 00 00 00 00 00 00
can0 311 [8] FF FF FF FA FF FF FF FF
can0 241 [8] AA 00 00 00 00 00 00 4B
can0 264 [8] 01 12 00 1F 1E 00 00 00
can0 231 [8] 00 00 00 02 01 00 00 00
can0 251 [8] 00 00 00 01 00 00 00 00
can0 252 [8] 00 00 00 00 00 00 00 00
can0 253 [8] 00 00 00 01 00 00 00 00
can0 254 [8] 00 00 00 00 00 00 00 00
can0 221 [8] 00 00 00 00 00 00 00 00
can0 311 [8] FF FF FF FA FF FF FF FF
can0 241 [8] AA 00 00 00 00 00 00 4C
can0 251 [8] 00 00 00 01 00 00 00 00
can0 252 [8] 00 00 00 00 00 00 00 00
can0 253 [8] 00 00 00 01 00 00 00 00
can0 254 [8] 00 00 00 00 00 00 00 00
can0 221 [8] 00 00 00 00 00 00 00 00
can0 311 [8] FF FF FF FA FF FF FF FF
can0 241 [8] AA 00 00 00 00 00 00 4D
can0 251 [8] 00 00 00 01 00 00 00 00
can0 252 [8] 00 00 00 00 00 00 00 00
can0 253 [8] 00 00 00 01 00 00 00 00
can0 254 [8] 00 00 00 00 00 00 00 00
can0 221 [8] 00 00 00 00 00 00 00 00
can0 311 [8] FF FF FF FA FF FF FF FF
can0 241 [8] AA 00 00 00 00 00 00 4E
```

## 3.6.2 SCOUT MINI ROS Package

scout\_bringup: Start up the mobile base node.

scout\_base: Control and monitor package based on ugv\_sdk.

scout\_description: scout\_min URDF model, available for simulation of customized robots equipped with sensors .

scout\_msgs: scout\_mini message format definition.

```
/home/agilex/catkin_ws/src/scout_base/scout_base/launch/scout_mini_base.launch http://localhost:11311
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://xavier:41683/

SUMMARY
=====

PARAMETERS
* /roscdistro: melodic
* /rosversion: 1.14.10
* /scout_base_node/base_frame: base_link
* /scout_base_node/is_scout_mini: True
* /scout_base_node/odom_frame: odom
* /scout_base_node/odom_topic_name: odom
* /scout_base_node/port_name: can0
* /scout_base_node/simulated_robot: False

NODES
/
  scout_base_node (scout_base/scout_base_node)

auto-starting new master
process[roscdistro]: started with pid [11302]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 953e1c90-2f6b-11ed-b7eb-70cd0d910208
process[rosout-1]: started with pid [11317]
started core service [/rosout]
process[scout_base_node-2]: started with pid [11324]
Working as scout mini: 1
Start listening to port: can0
[ INFO ] [1662637465.185996843]: Using CAN bus to talk with the robot
```

### 3.6.3 IMU sensor(Optional)

imu\_launch: launch imu function package

```
roslaunch imu_launch imu_msg.launch
```

To view IMU data, enter in the terminal:

```
rostopic echo /imu/data_raw
```

```
agilex@xavier: ~
y: 0.0
z: 0.0
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
  x: 0.0980598889291
  y: -0.0244327200577
  z: 9.7587864399
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
header:
  seq: 2332
  stamp:
    secs: 1662637368
    nsecs: 131232262
  frame_id: "imu_link"
orientation:
  x: -0.00357459671795
  y: -0.003524111351
  z: -0.000401197234169
  w: 0.999987363815
orientation_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: 0.0
  y: 0.0
  z: 0.0
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
  x: 0.0980598889291
  y: -0.0244327200577
  z: 9.7587864399
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
agilex@xavier:~$
```

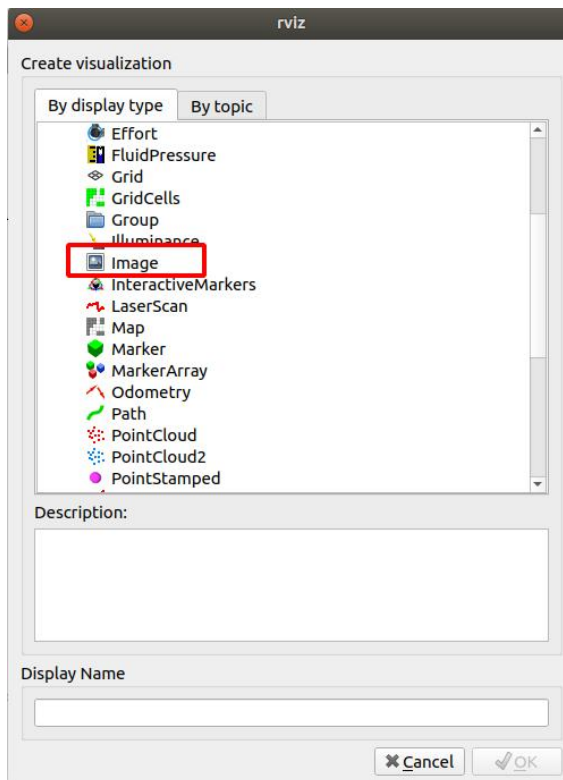
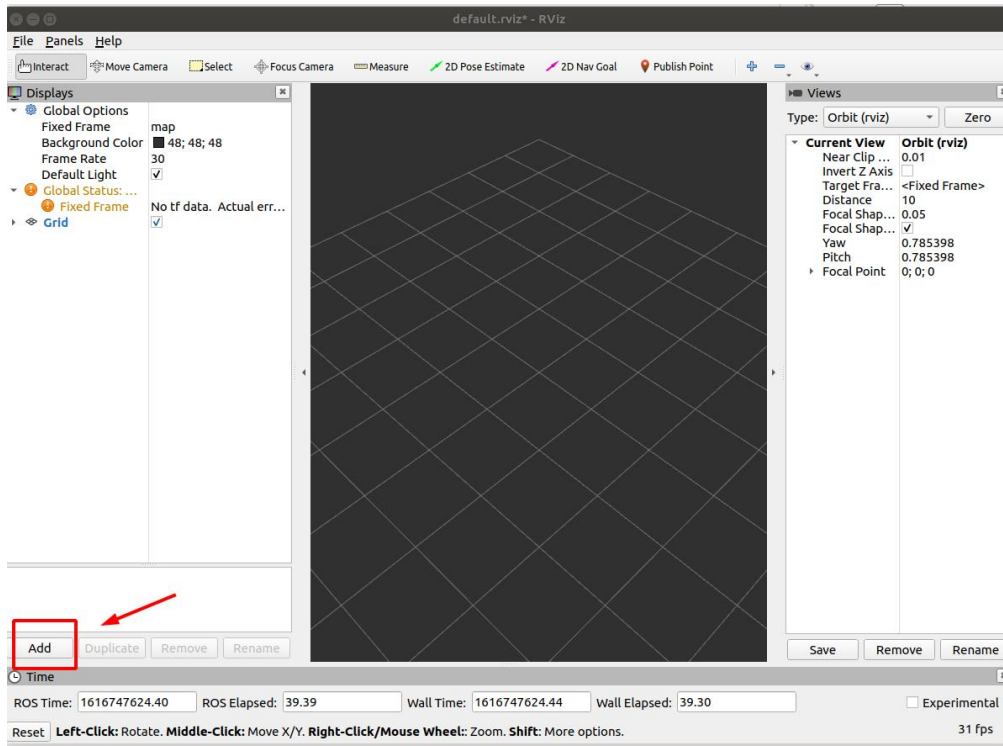
### 3.6.3 RealSense D435 node ROS Package and RVIZ visualization

Realsense2\_camera: Set up and start the depth camera function package, and you can use the rviz visualization tool to view color maps, depth maps, dense point clouds, etc.

#### Get the rgb

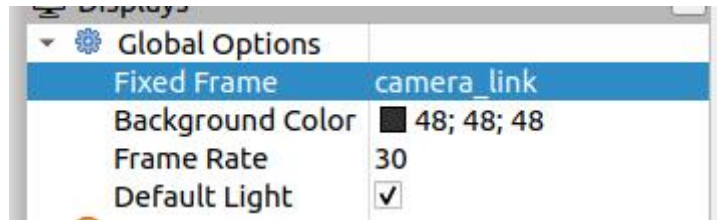
```
roslaunch realsense2_camera rs_camera.launch
```

Enter rviz in the terminal to open rviz tool, click the add to add image component.

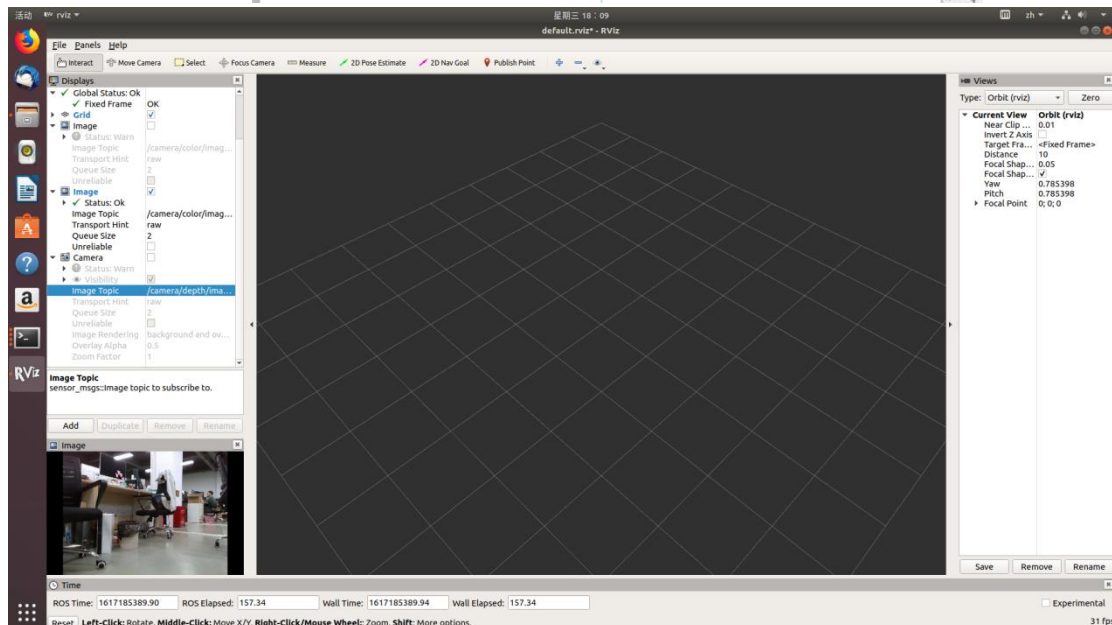
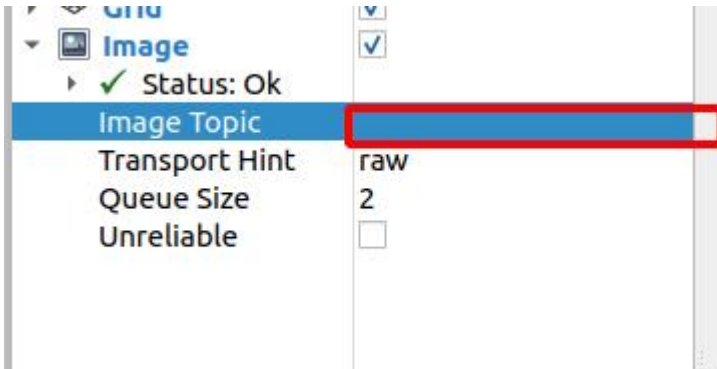


Fixed frame: choose camera\_link.





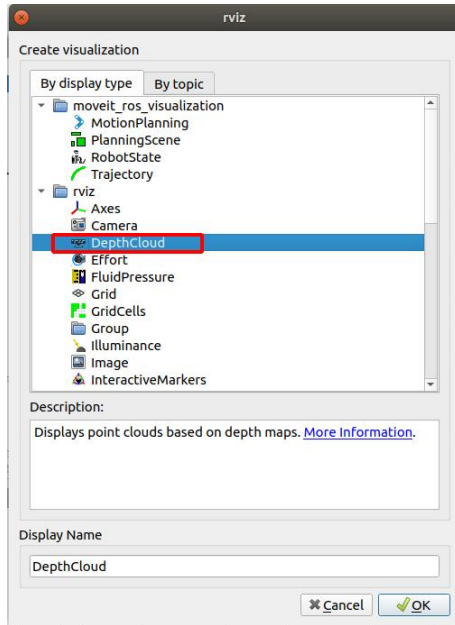
Fills in the corresponding topic in image component to get the rgb picture.



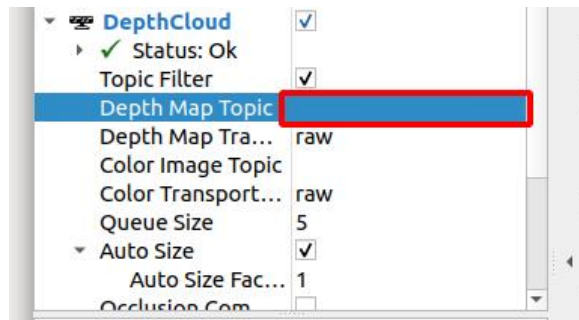
### Get the rgb + Depth map

```
roslaunch realsense2_camera rs_camera.launch
```

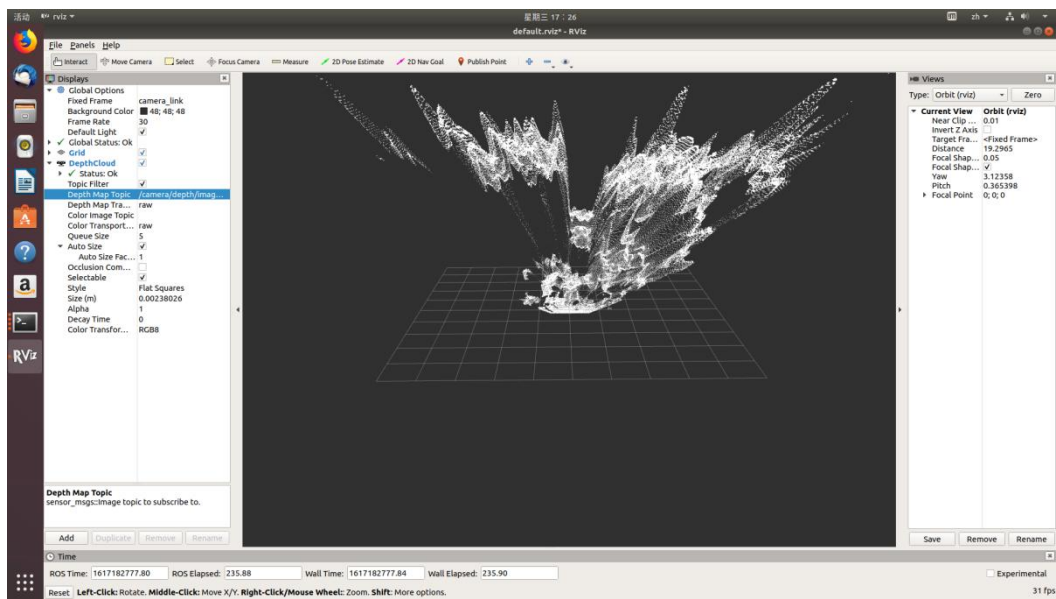
Open rviz, click the add to add DepthCloud component.



Fixed frame : choose camera\_link, choose the corresponding topic in the Depth Map topic.



The depth map is shown below:



## 3.6.5 RS-Helios-16P

rslidar\_sdk: driver package for lidar.

### 3.6.5.1 Set the industrial computer IP

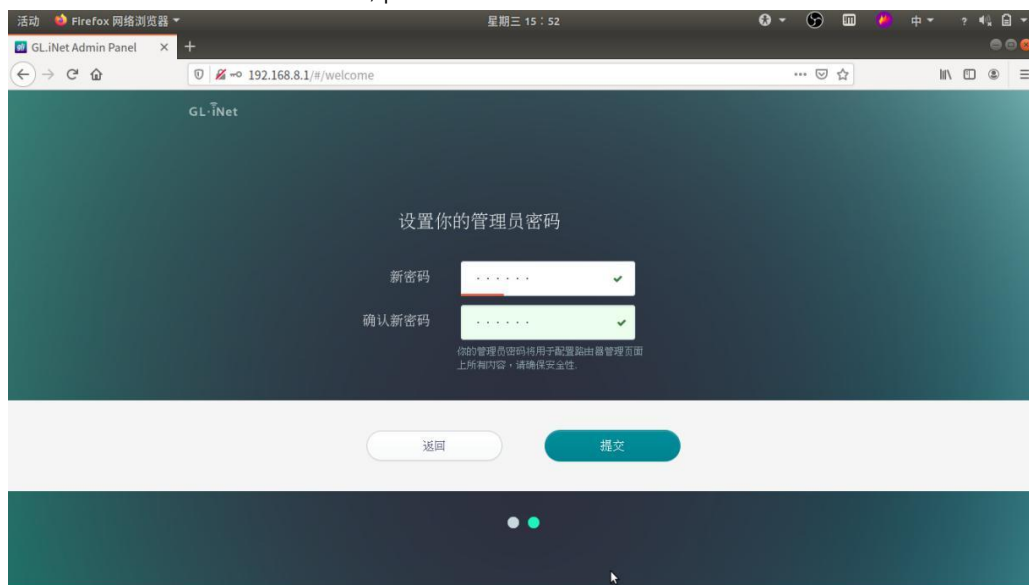
Open the browser and enter in the browser search bar:

192.168.1.1

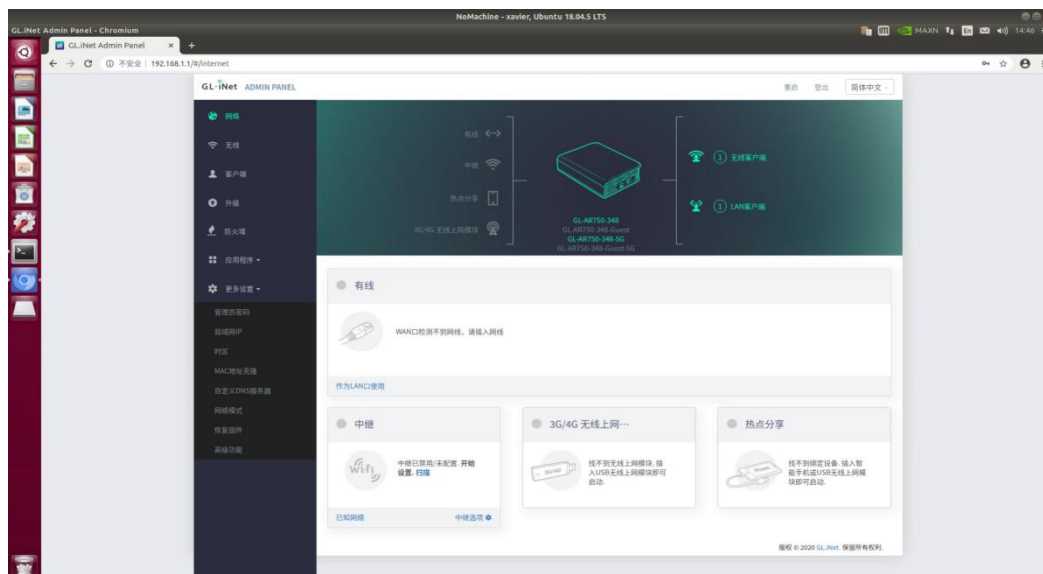
Enter the router management web page.

The password is uniformly set to 12345678.

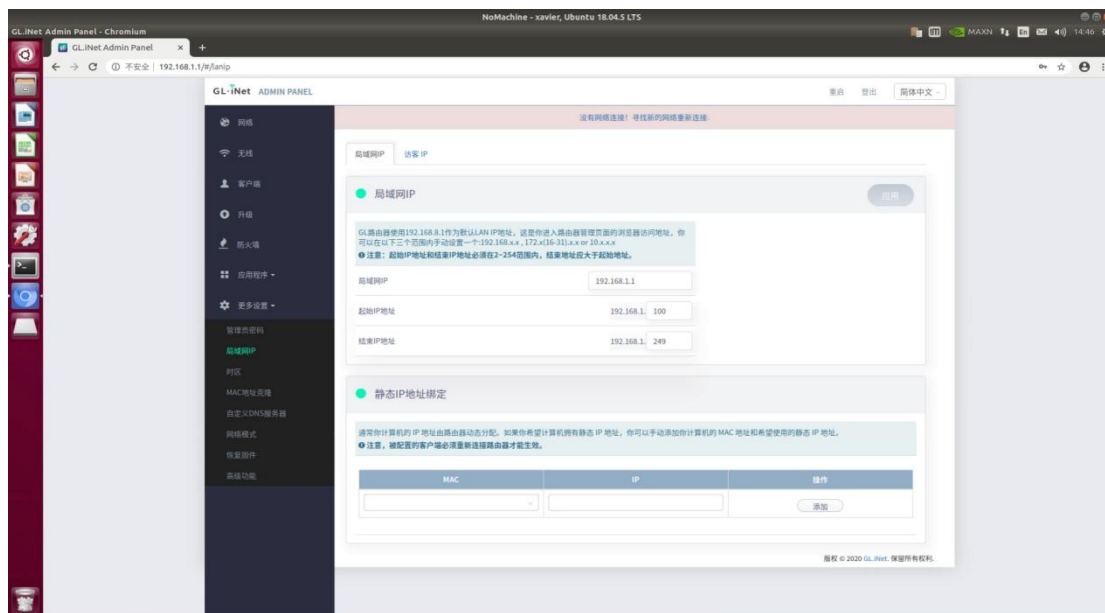
Note: If the router has been reset, please enter 192.168.8.1 in the browser.



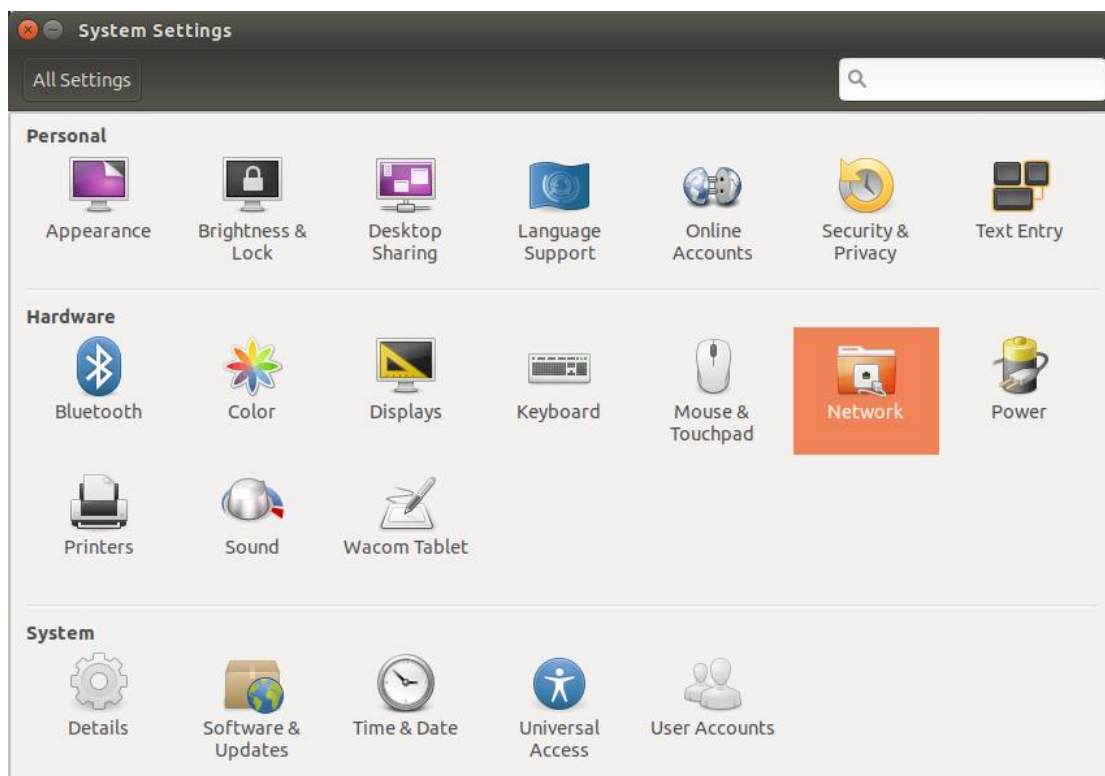
Enter the router management interface and click More Settings on the left.



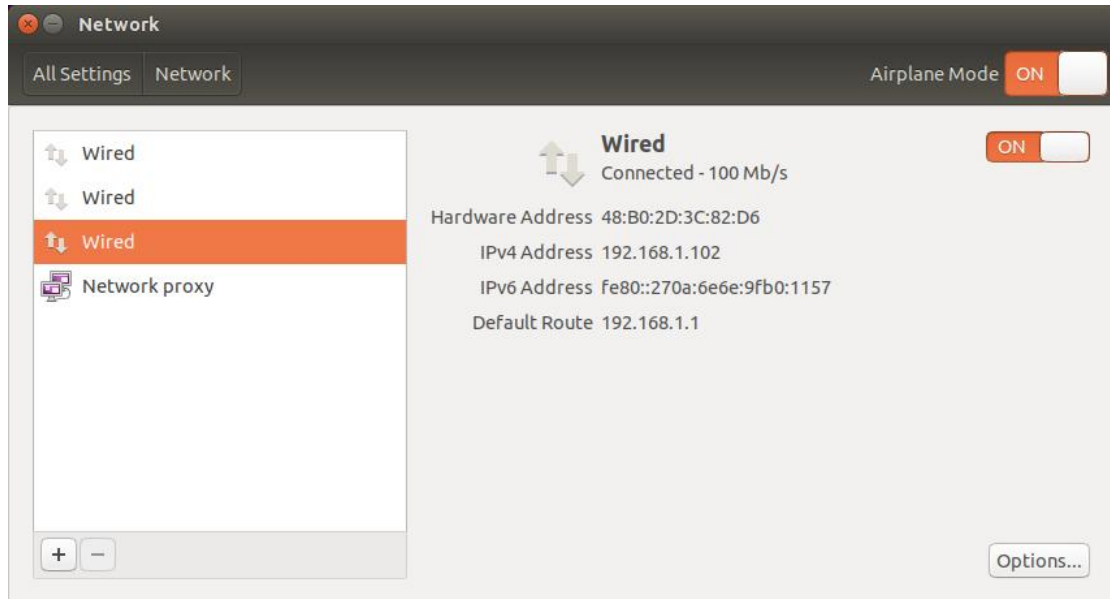
Continue to select the LAN IP and change the LAN IP to 192.168.1.1



Then open computer settings and select Network.



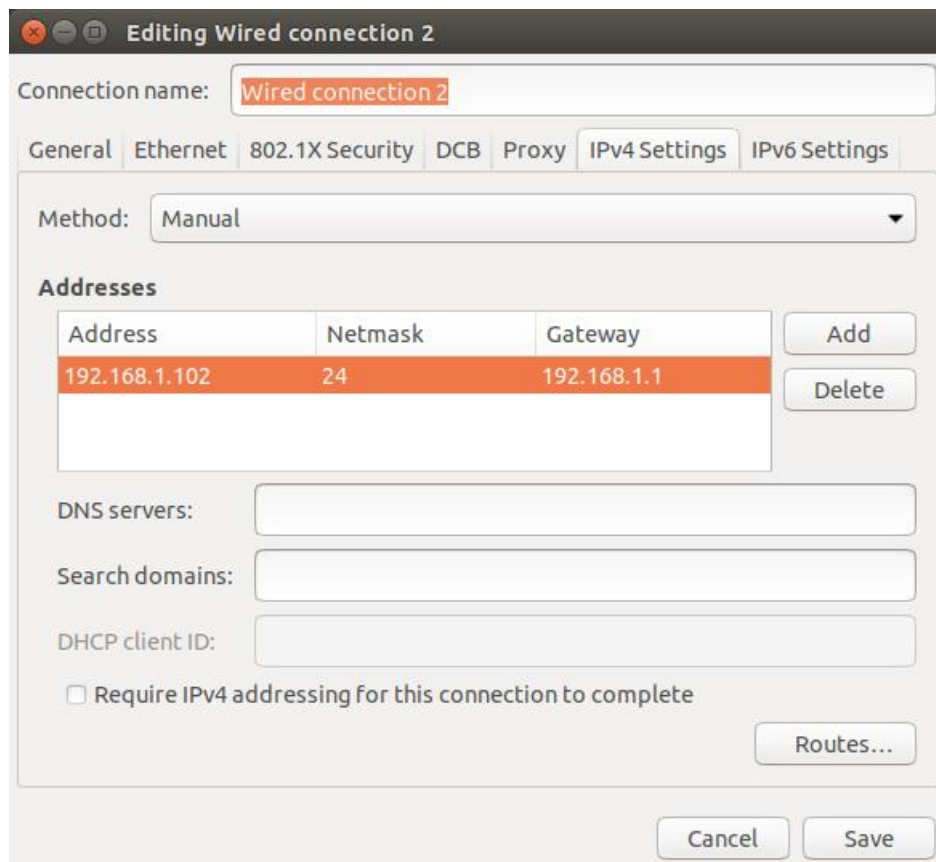
Select the network to connect to, usually the third one, then click options to enter the network settings



Follow these steps

Select IPv4; click the subscript on the right side of Method and select Manual; click add and enter 192.168.1.102 in Address.

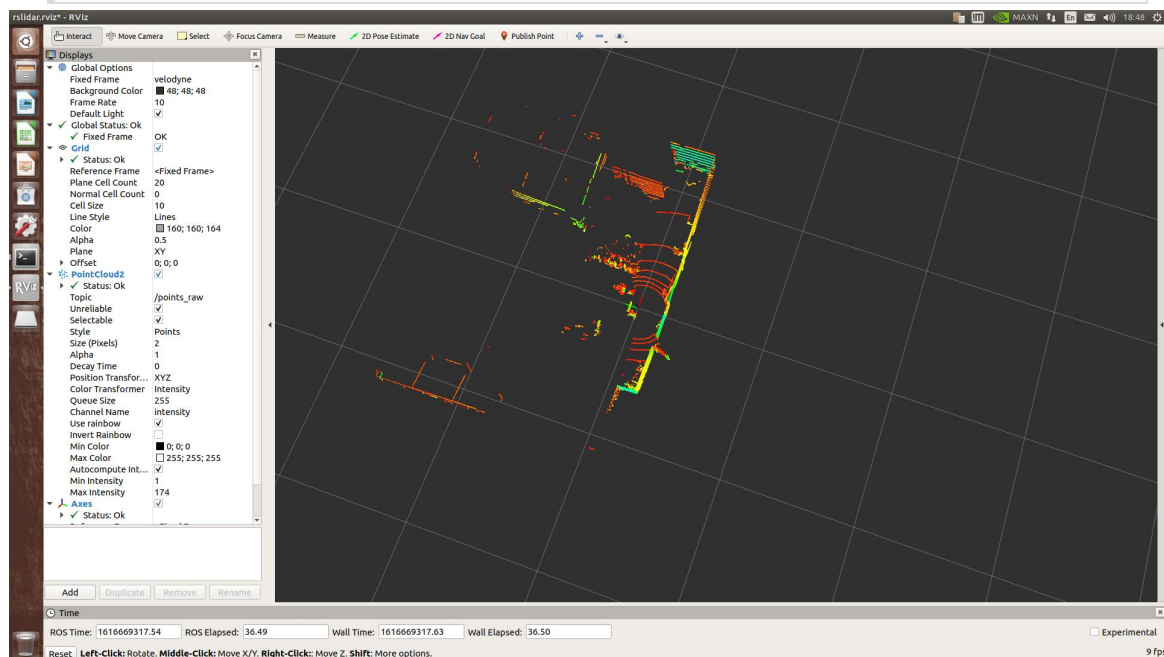
Enter 24 in Netmask, enter 192.168.1.1 in Gateway, and click save to save. The final effect is as follows:



### 3.6.5.2 Start RS-Helios-16P

Start the lidar and publish the coordinate transformation of base\_link-><laser\_link>

```
roslaunch scout_bringup open_rslidar.launch
```



## 3.7 Navigation and positioning based on Gmapping open source architecture

### 3.7.1 Autonomous Navigation Practice of Lidar

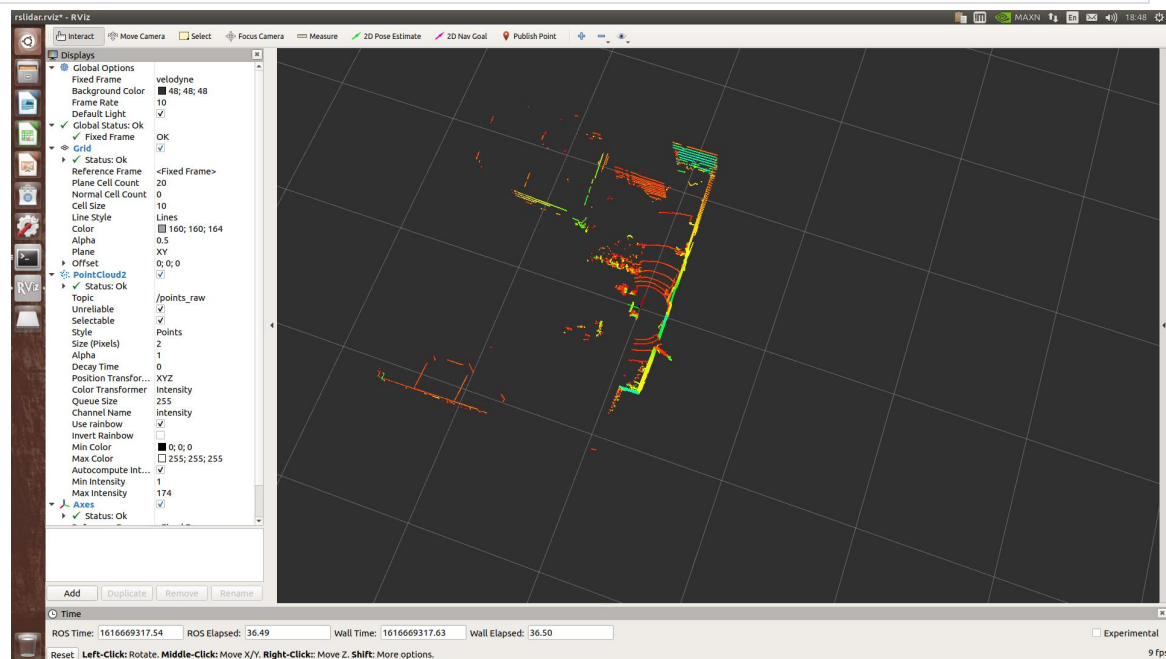
The RS-Helios-16P radar is a 16-beam laser radar with a 2° increment, covering 15° in both the upward and downward directions. As the RS-Helios-16P directly outputs point cloud data, it includes laser data from all 16 beams. Gmapping mapping only requires data from one beam, so it is necessary to compress the point cloud data. At this point, you can use the ROS package provided by VLP to compress and output point cloud data to /scan. Alternatively, you can utilize the pointcloud\_to\_laserscan package that we provide for compressing point cloud data. Due to the scanning angles of 3D laser radar, the scanning blind spots are fewer compared to 2D radar. Once the scan data is obtained, it can be fed to Gmapping for mapping. Provide a relatively reliable odometer information and correct coordinate transformations, and Gmapping mapping can be performed.

#### 3.7.1.1 Autonomous navigation practice of lidar

Slam mapping practice:

(1) Start the lidar and publish the coordinate transformation of base\_link-><laser\_link>

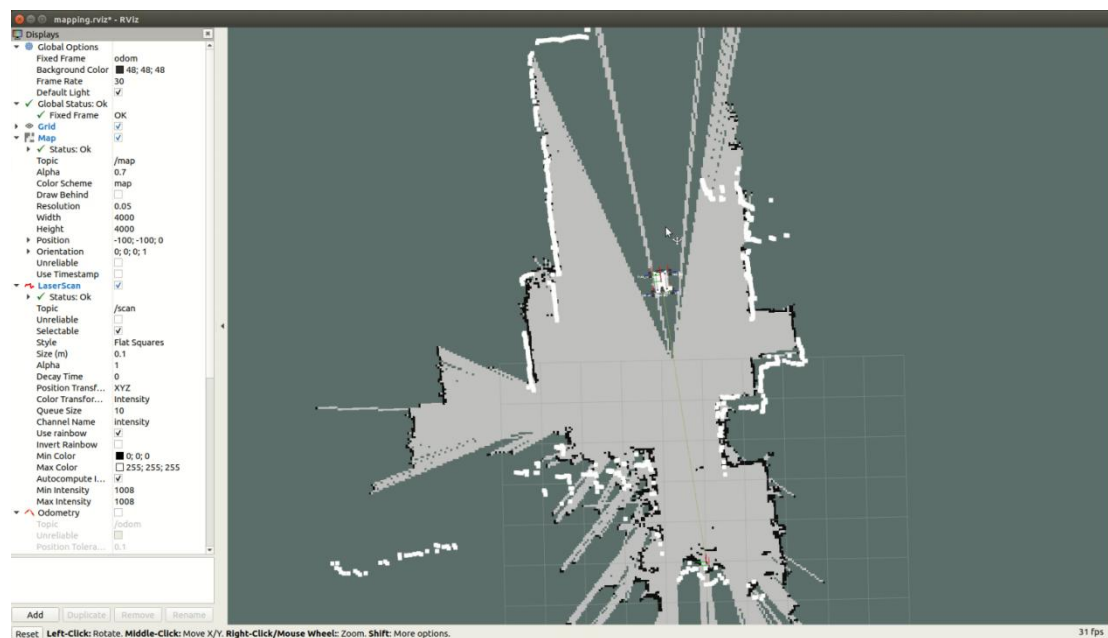
```
roslaunch scout_bringup open_rslidar.launch
```



(2) Launch gmapping node

Gmapping Algorithm:

```
roslaunch scout_bringup gmapping.launch
```



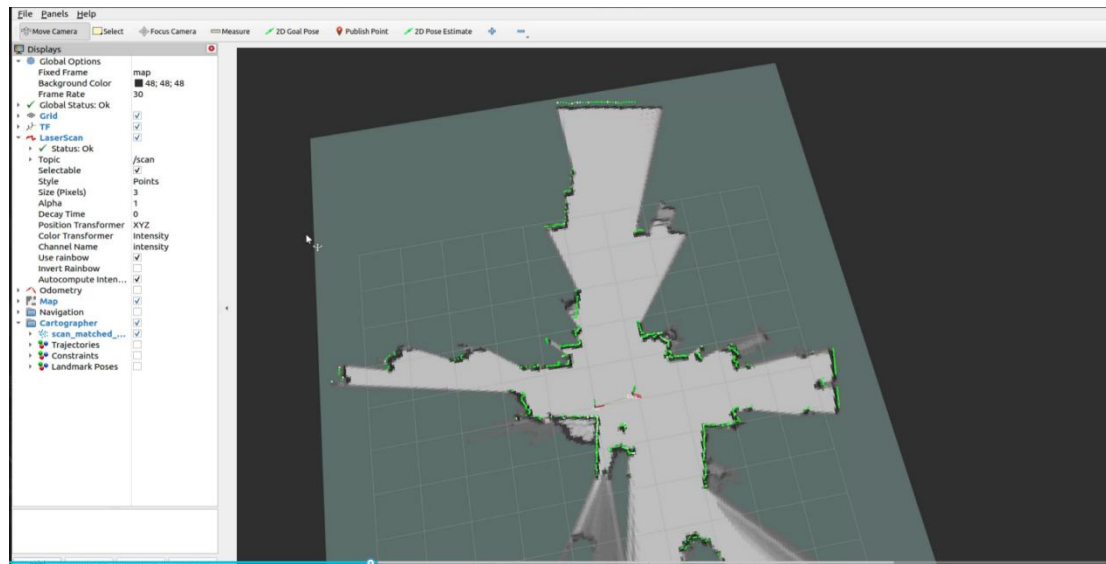
Use the joystick to control the robot to move around the scene. After building the map, save the map to the specified directory (usually to 'description'). The name of this map is 'map'. Please be careful not to repeat the map name.

```
roslaunch map_server map_saver -f ~/catkin_ws/src/scout_base/scout_description/maps/map
```

**Note:** The last 'map' is the map name. Any English characters can be used.

## Cartographer Algorithm

```
roslaunch scout_bringup scout_cartographer.launch
```



Use the joytick to control the robot to move around the scene. After building the map, save the map to the specified directory (usually to 'description'). The name of this map is 'map'. Please be careful not to repeat the map name.

```
roslaunch map_server map_saver -f ~/catkin_ws/src/scout_base/scout_description/maps/map
```

## Autonomous navigation practice:

(1) Start the lidar and publish the coordinate transformation of base\_link-> <laser\_link>.

```
roslaunch scout_bringup open_rslidar.launch
```

(2) Start move\_base navigation.

```
roslaunch scout_bringup navigation_4wd.launch
```

**Note:** If you need to customize the opened map, please open the navigation.launch file to modify the parameters, as shown in the figure below. Please modify the marked horizontal line to the name of the map you want to open.



```

<?xml version="1"?>
<!--
Simulate a carlike robot with the teb_local_planner in stage:
- stage
- map_server
- move_base
- static map
- amcl
- rviz view
-->
<launch>
  <!-- ***** Navigation ***** -->
  <node pkg="move_base" type="move_base" respawn="false" name="move_base" output="screen">
    <roscppparam file="$(find scout_description)/param/4wd/costmap_common_params.yaml" command="load" ns="global_costmap" />
    <roscppparam file="$(find scout_description)/param/4wd/costmap_common_params.yaml" command="load" ns="local_costmap" />
    <roscppparam file="$(find scout_description)/param/4wd/local_costmap_params.yaml" command="load" />
    <roscppparam file="$(find scout_description)/param/4wd/global_costmap_params.yaml" command="load" />
    <roscppparam file="$(find scout_description)/param/4wd/base_local_planner_params.yaml" command="load" />
    <roscppparam file="$(find scout_description)/param/4wd/move_base_params.yaml" command="load" />
  </node>

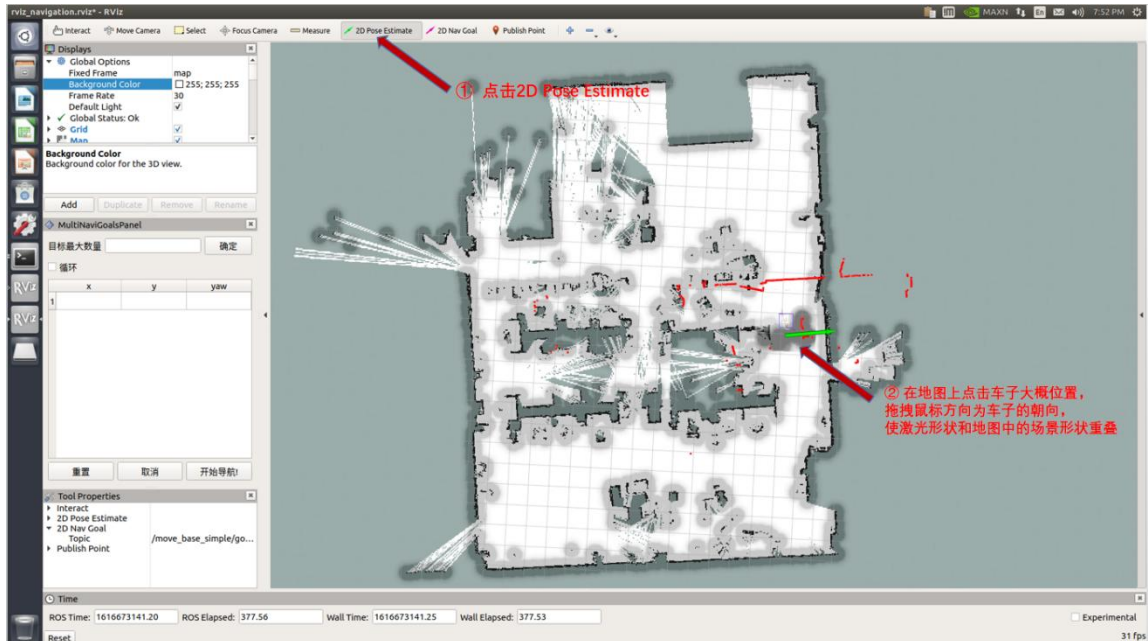
  <!-- ***** Maps ***** -->
  <node name="map_server" pkg="map_server" type="map_server" args="$(find scout_description)/maps/map2.yaml" output="screen">
    <!-- <node name="map_server" pkg="map_server" type="map_server" args="/home/nvidia/map/2.yaml" output="screen"-->
    <param name="frame_id" value="map" />
  </node>

  <node pkg="amcl" type="amcl" name="amcl" output="screen">
    <roscppparam file="$(find scout_description)/param/amcl_params.yaml" command="load" />
    <param name="initial_pose_x" value="0" />
    <param name="initial_pose_y" value="0" />
    <param name="initial_pose_a" value="0" />
  </node>

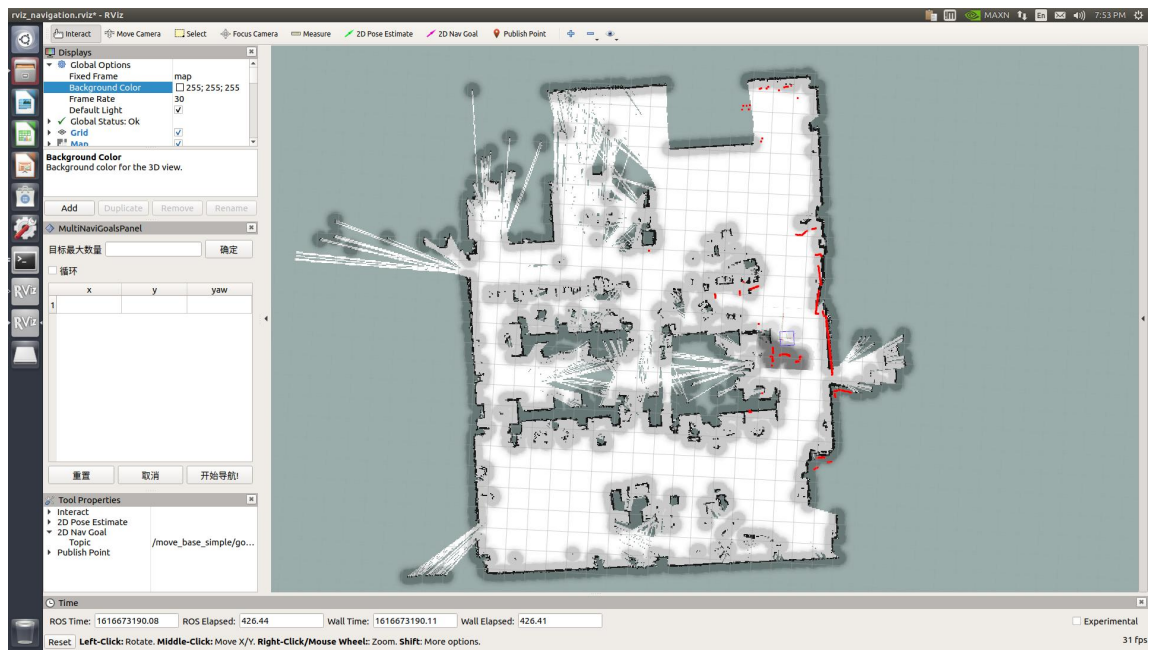
  <!-- ***** Visualisation ***** -->
  <node name="car_rviz" pkg="rviz" type="rviz" args="-d $(find scout_description)/rviz/rviz_navigation.rviz">
  </node>
  <include file="$(find scout_bringup)/launch/scout_velocity_smoother.launch"/>
  <include file="$(find scout_base)/launch/scout_mini_base.launch">
    <arg name="port_name" default="can0" />
    <arg name="simulated_robot" default="false" />
  </include>
</launch>

```

(3) Correct the actual position of the chassis in the scene on the map displayed in rviz by publishing an approximate position and use a joystick to correct the chassis rotation. Correction is completed when the laser shape overlaps the scene shape in the map.



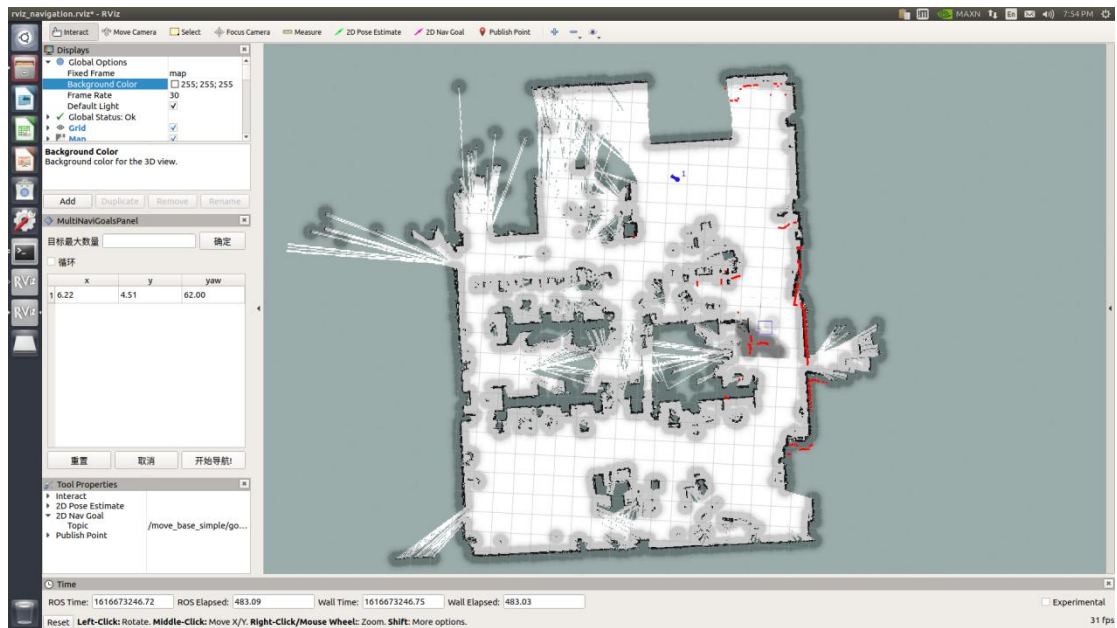
After setting the positioning, the laser shape and the scene shape in the map have basically overlapped, and the correction is completed. As shown below



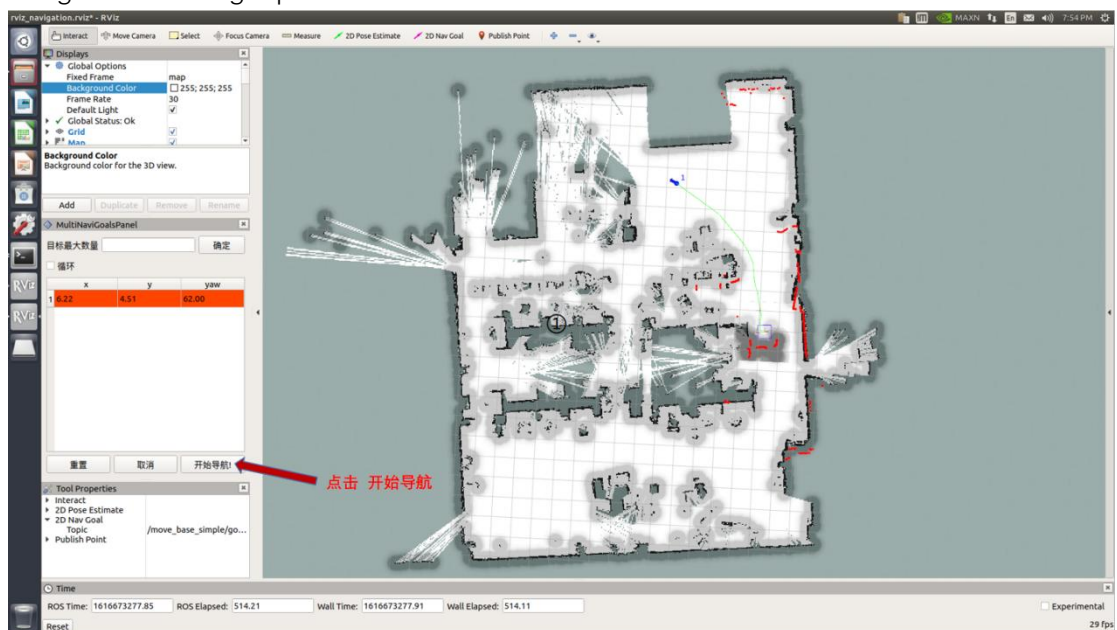
(4) Set the target point on the multi-point navigation plug-in on the left column, click to start navigation, and switch the controller to command control mode.



The target point is generated, as the picture shown below.



Click to start navigation, the map has generated a path (in green), and it will automatically navigate to the target point.



## 3.8 Navigation and positioning on vision and laser

### 3.8.1 rtamap algorithm introduction

The rtamap algorithm provides an appearance-based positioning and composition solution that is independent of time and scale. Aimed at solving online closed-loop detection problems in large environments. The idea of the solution is that in order to meet some limitations of real-time performance, closed-loop detection only uses a limited

number of anchor points, and at the same time can access the anchor points of the entire map when needed.

### 3.8.2 rtabmap mapping algorithm

(1) Launch the radar. Enter the command in the terminal:

```
roslaunch scout_bringup open_rslidar.launch
```

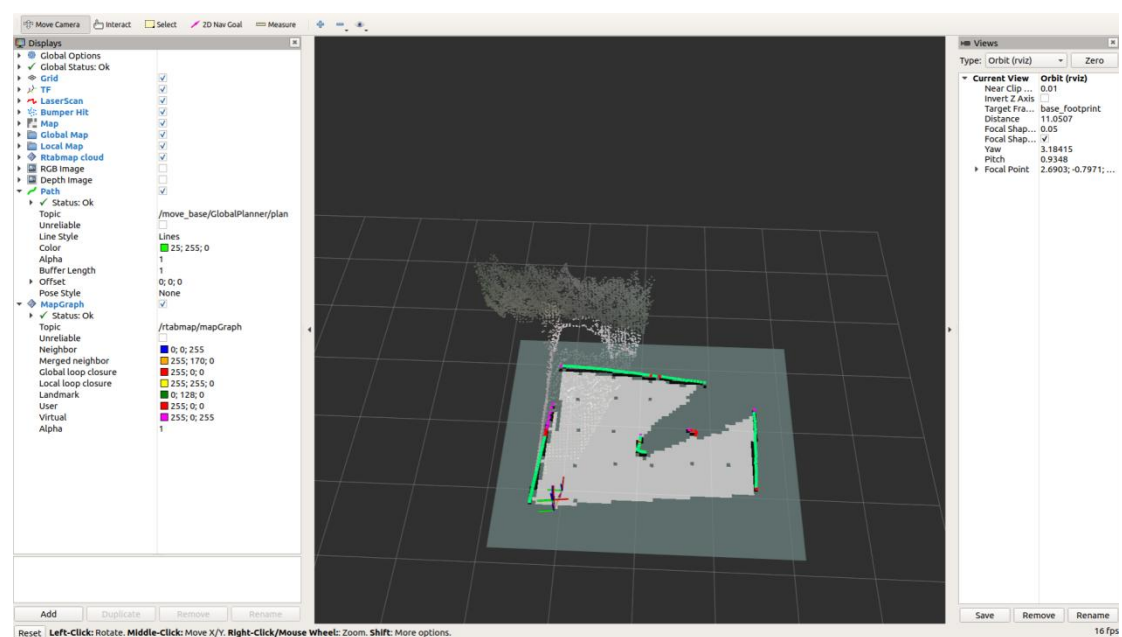
(2) Launch the realsense. Enter the command in the terminal:

```
roslaunch realsense2_camera rs_aligned_depth.launch
```

(3) Launch the rtamap algorithm mapping mode. Enter the command in the terminal:

```
roslaunch scout_bringup rtabmapping.launch
```

When the rviz interface shown is in the picture below, the rtabmap mapping mode starts successfully.



After building the map, you can terminate the program directly and the built map will be automatically saved in the `.ros` file in the home directory with the file name `rtabmap.db`. The `.ros` folder is a hidden folder and needs to be displayed through the `Ctrl+h` command.

### 3.8.3 rtabmap navigation algorithm

(1) Launch the LiDAR. Enter the command in the terminal:

```
roslaunch scout_bringup open_rslidar.launch
```

(2) Launch the realsense. Enter the command in the terminal.

```
roslaunch realsense2_camera rs_aligned_depth.launch
```

(3) Launch the mapping mode. Enter the command in the terminal:

```
roslaunch scout_bringup rtabmapping.launch localization:=true
```

(4) Launch move\_base. Enter the command in the terminal:

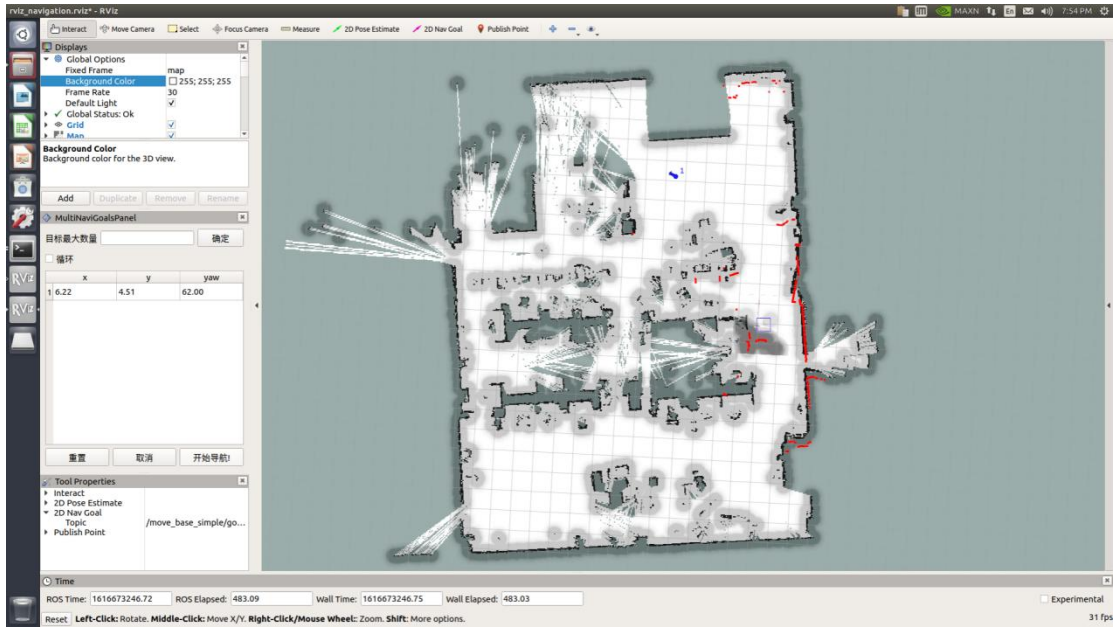
```
roslaunch scout_bringup scout_rtab_navigation.launch
```

Because we use visual positioning, there is no need to make corrections when using rtabmap navigation. You can directly start setting the target point for navigation. The operation steps are shown in the figure.

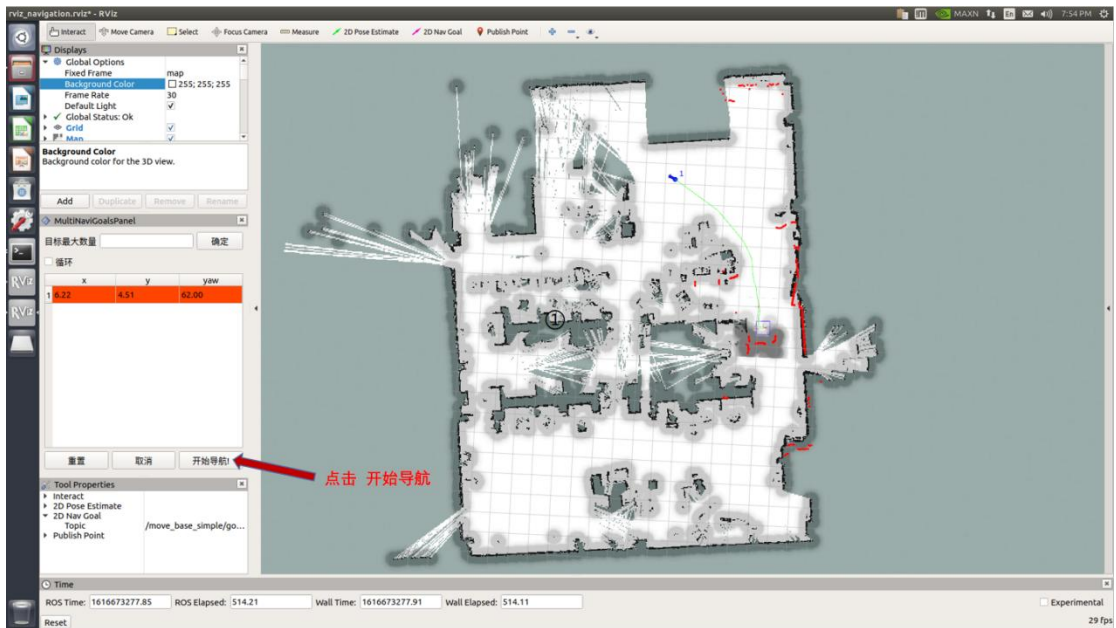
Set the target point on the multi-point navigation plug-in on the left column, click to start navigation, and switch the handle to command control mode.



The target has been generated.



Click to start navigation. The map has generated a path (green) and will automatically navigate to the target point.



## 3.9 Visual Application

### 3.9.1 Traffic lights identification

#### 3.9.1.1 Function Introduction

After detecting the traffic light target through darknet\_ros, it is necessary to identify the traffic light color and position it in the three-dimensional space to generate the positional relationship between the object and the camera. This method can only achieve the identification and positioning of traffic lights, but cannot obtain the traffic light posture. A depth camera is required, and its recognition distance depends on the depth camera range.

#### 3.9.1.2 Operation

Launch the realense depth camera. Enter the command in the terminal:

```
roslaunch realsense2_camera rs_camera.launch
```

Launch the yolo\_v3. Enter the command in the terminal:

```
roslaunch darknet_ros yolo_v3_tiny.launch
```

Launch the traffic lights identification function:

```
roslaunch scout_deeplearning traffic_light_located.launch
```

### 3.9.2 QR code Tracking

#### 3.9.2.1 Function Introduction

After QR code recognition through ar\_track\_alvar, the location information of the QR code is obtained and the TF relationship between the QR code and the camera is generated.

#### 3.9.2.2 Operation

Launch realsense depth camera. Enter the command in the terminal:

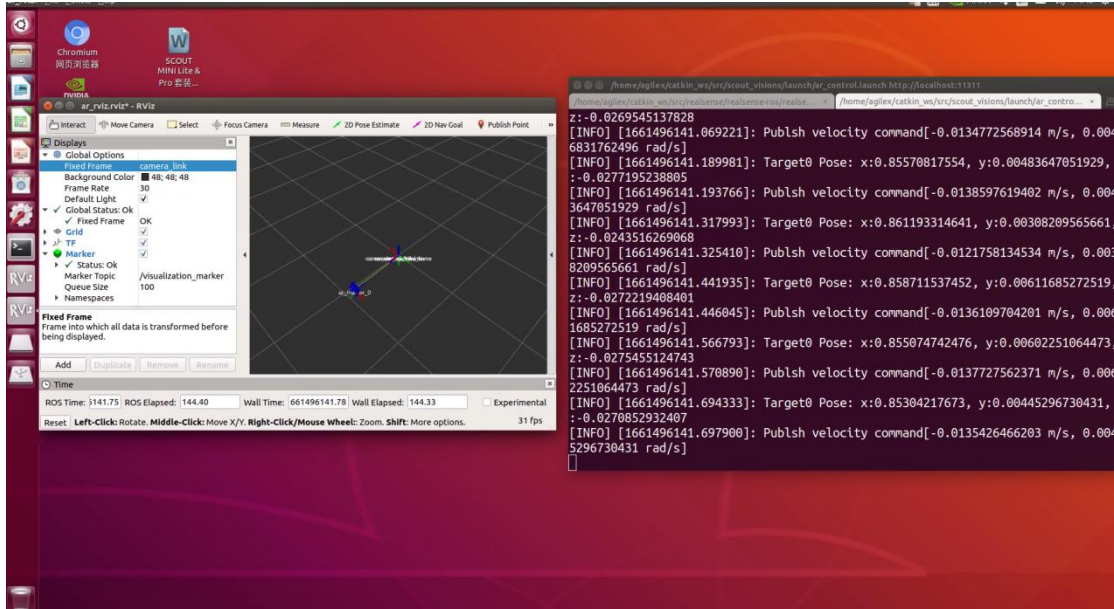
```
roslaunch realsense2_camera rs_camera.launch
```

Launch the QR code identification. Enter the command in the terminal:

```
roslaunch scout_visions ar_code.launch
```

Launch the tracking function. Enter the command in the terminal:

```
roslaunch scout_visions ar_control.launch
```





# Common problems

## 1. CAN0 cannot be recognized

- 1) It may be a problem with the USB interface on the computer and needs to be replaced.
- 2) Or maybe there are no gs\_usb drivers. Take the TX2 for example and add the gs\_usb module.

```
git clone https://github.com/HubertD/socketcan_gs_usb.git
```

```
cd socketcan_gs_usb
```

```
make
```

If there is error during 'make', please recompile the kernel.

```
cd /lib/modules/4.4.38-tegra/
```

```
cd build
```

```
sudo make
```

Go back to the socketcan\_gs\_usb directory, execute make, and add the gs\_usb.ko file after successful compilation.

```
sudo modprobe can_dev can insmod gs_usb.ko
```

```
ifconfig -a
```

There is one more can2, added successfully!

## 2. There is no data returning back after executing candump can0

- 1) There may be a problem with the microUSB cable and need to replace it with a new one.
- 2) The high and low connections of the CAN are reversed. The yellow one should be connected to can\_H and the blue one should be can\_L. If the connection is reversed, it needs to be replaced.
- 3) Virtual machine. If you are using a virtual machine, you will not be able to receive data.

- 4) USB to CAN module fail.
- 5) `gs_usb` module is not enabled.

### 3. If 'device or resource busy' occurs

It means that it may be enabled repeatedly. In this case, you need to unplug and plug the USB port again to solve the problem.

### 4. Recognized as other serial number. For example: `can2`

You can change all '`can0`' in the code to '`can2`'

### 5. If 'Network is down' occurs

The Can may be not enabled. Execute the following command:

```
roslaunch scout_bringup setup_can2usb.bash
```

### 6. The `asio.hpp` problem

Execute the following command:

```
sudo apt-get install libasio-dev
```

### 7. '`pcl_ros`' occurs when compiling

Execute the following command:

```
sudo apt-get install ros-melodic-pcl-ros
```

### 8. Errors occur when launching LiDAR

1. ERROR: "base\_link" passed to lookupTransform argument source\_frame does not exist. The error is caused by TF not being fully started. It will be automatically corrected after it is fully started.

2. 'Waiting for laser\_scans...'

- (1) Check the whether the cable of LiDAR is correctly connected

- (2) Check whether the Router's network segment is:

```
192.168.1.1
```

- (3) Check the ip address of IPC. The default address is

```
192.168.1.102
```

# Appendix

## Appendix1 Basic Operating Commands

### 2.1 Directory operating commands

#### (1) Directory switch: cd

- ① cd / switch to the root directory
- ② cd /usr switch to the usr directory under the root directory
- ③ cd ../ switch to the upper level directory or cd ..
- ④ cd ~ switch to the home directory
- ⑤ cd - switch to the last visited directory

#### (2) Directory view: ls

- ① ls: view all directories and files in the current directory
- ② ls -a: view all directories and files in the current directory (including hidden files)
- ③ ls -l or ll: list view all directories and files in the current directory (list view, which displays more information)
- ④ ls /dir: view all directories and files in the specified directory, like: ls /usr

#### (3) Create directory: mkdir

- ① mkdir aaa: create a directory named aaa in the current directory
- ② mkdir /usr/aa: create a directory named aaa in the specified directory

#### (4) Show hidden directory: Ctrl+h

In the folder, you can display the hidden folders in the folder by Ctrl+h

#### (5) Terminate program: Ctrl+c

Enter the command Ctrl+c in the terminal to forcefully terminate the program being executed

### 2.2 ROS commonly used commands

#### (1) Compile command: catkin\_make

Used to compile packages in the entire workspace

(2) Initialize workspace: `catkin_init_workspace`

Used to initialize the workspace when creating the workspace for the first time

(3) Create package: `catkin_create_pkg`

Used to create a package, and its syntax is:

```
catkin_create_pkg <package_name> [depend1] [depend2] [depend3]...
```

#### (4) Node running command

① `roslaunch` is used to run the `.launch` files. In the launch files, the `.cpp` files and the `.py` files can be called at the same time, and its syntax is:

```
roslaunch package_name node_name
```

② `roslaunch` is used to run `.launch` files. In the launch files, the `.cpp` files and the `.py` files can be called at the same time, and its syntax is:

```
roslaunch package_name node_name
```

## Appendix2 Parameter Configuration of Navigation Package

If you need to try to debug the parameters in the package yourself, you can refer to the following list.

### 7.1 Configurable parameters in the gmapping package

Parameter	Type	Default	Description
<code>~throttle_scans</code>	int	1	The scan data threshold to be processed; the default is to process 1 scan data at a time (it can be set larger to skip some scan data)

Parameter	Type	Default	Description
~base_frame	string	base_link	Robot base coordinate system
~map_frame	string	map	Map coordinate system
~odom_frame	string	odom	Odometer coordinate system
~map_update_interval	float	5.0	Map update frequency
~maxUrange	float	80	Detect the maximum available range, that is, the range that the beam can reach
~sigma	float	0.05	Standard deviation of endpoint matching
~kernelSize	int	1	Used to find the corresponding kernel size
~lstep	float	0.05	Translation optimization step
~astep	float	0.05	Rotation optimization step
~iterations	int	5	Scan matching iterations
~lsigma	float	0.075	Laser standard deviation for likelihood calculation
~ogain	float	3.0	Used for smooth resampling effect during likelihood calculation

Parameter	Type	Default	Description
~lskip	int	0	The number of beams skipped in each scan.
~minimumScore	float	0.0	The lowest value of the scan matching result
~srr	float	0.1	The mileage error during translation as a translation function (rho/rho)
~srt	float	0.2	The mileage error during translation as a rotation function (rho/theta)
~str	float	0.1	The mileage error during rotation as a translation function(theta/rho)
~stt	float	0.2	The mileage error during rotation as a rotation function (theta/theta)
~linearUpdate	float	1.0	The robot translates a certain distance and processes the laser data once
~angularUpdate	float	0.5	The robot rotates a certain distance and processes the laser

Parameter	Type	Default	Description
			data once
~temporalUpdate	float	-1.0	If the latest scan processing is slower than the update, one scan is processed. Turn off time-based updates when the value is negative.
~resampleThreshold	float	0.5	Resampling threshold based on Neff
~particles	int	30	Number of particles in the filter
~xmin	float	-100.0	The initial minimum size of the map in the x direction
~ymin	float	-100.0	The initial minimum size of the map in the y direction
~xmax	float	100.0	The initial maximum size of the map in the x direction
~ymax	float	100.0	The initial maximum size of the map in the y direction
~delta	float	0.05	Map resolution

Parameter	Type	Default	Description
~llsamplerange	float	0.01	The translation sampling distance of likelihood calculation
~llsamplestep	float	0.01	The translation sampling step of likelihood calculation
~lasamplerange	float	0.005	The angle sampling distance of likelihood calculation
~lasamplestep	float	0.005	The angle sampling step of likelihood calculation
~transform_publish_period	float	0.05	TF transform publishing period
~occ_thresh	float	0.25	The threshold of raster map occupancy rate
~maxRange	float	—	The maximum range of sensor

## 7.2 Configurable parameters in the cartographer package

Parameter	Default	Analysis
map_frame	map	The ID of the ROS coordinate system used to publish submaps, the parent coordinate system of the pose, usually "map".



Parameter	Default	Analysis
tracking_frame	base_footprint	The ID of the ROS coordinate system tracked by the SLAM algorithm. If IMU is used, its coordinate system should be used, usually "imu_link".
published_frame	odom	The ID of the ROS coordinate system used to publish the pose sub-coordinate system, like the "odom" coordinate system. If an "odom" coordinate system is provided by different parts of the system, in this case, the "odom" pose in the map_frame will be published. Otherwise, it may be appropriate to set it to "base_link".
odom_frame	odom	It is enabled when provide_odom_frame is true. The coordinate system is used to publish local SLAM results

Parameter	Default	Analysis
		between published_frame and map_frame, usually "odom".
provide_odom_frame	true	If enabled, local, non-closed-loop, and continuous poses will be published as odom_frame in map_frame.
use_odometry	false	If enabled, subscribe to nav_msgs/Odometry messages on the "odom" topic. The mileage information will be provided, which is included in SLAM.
num_laser_scans	1	The number of laser scanning topics subscribed. Subscribe to sensor_msgs/LaserScan on the "scan" topic of one laser scanner or subscribe to the topics "scan_1", "scan_2", etc. on multiple laser scanners.

Parameter	Default	Analysis
num_multi_echo_laser_scans	0	The number of subscribed multi-echo laser scanning topics. Subscribe to sensor_msgs/MultiEchoLaserScan on the "echoes" topic of a laser scanner or subscribe to the topics "echoes_1", "echoes_2", etc. for multiple laser scanners.
num_subdivisions_per_laser_scan	1	The number of point clouds that divide each received (multi-echo) laser scan. The subdivision scan can cancel the scan acquired by the scan when the scanner is moving. There is a corresponding trajectory builder option to accumulate subdivision scans into the point cloud that will be used for scan matching.
num_point_clouds	0	The number of point cloud topics to be subscribed to. Subscribe to

Parameter	Default	Analysis
		sensor_msgs/PointCloud2 on the "points2" topic of a range finder or subscribe topics "points2_1", "points2_2", etc. for multiple range finders.
lookup_transform_timeout_sec	0.2	The timeout seconds of looking up and transforming with tf2.
submap_publish_period_sec	0.3	The period (in seconds) for publishing submaps, eg.0.3 seconds.
pose_publish_period_sec	5e-3	The period (in seconds) for publishing poses, eg. 5e-3, with a frequency of 200 Hz.
trajectory_publish_period_sec	30e-3	The period for publishing trajectory tag in seconds, eg. 30e-3, lasting 30 milliseconds.

### 7.3 Configurable parameters in the amcl package

Note: The parameter configuration files of the amcl package are: amcl\_param\_diff.yaml (the file is the amcl parameter file used in the four-wheel differential, omnidirectional wheel, and track motion modes), and amcl\_param.yaml (the file is the amcl parameter file used in the Ackermann motion mode) .

Parameter	Type	Default	Description
min_particles	int	100	The minimum number of particles allowed.
max_particles	int	5000	The maximum number of particles allowed.
kld_err	double	0.01	The maximum error between the true distribution and the estimated distribution.
kld_z	double	0.99	The upper normal quantile of (1-p), where p is the probability that the error on the estimated detuning will be less than kld_err.
update_min_d	double	0.2m	A translation movement needs to be performed before performing the filter update.
update_min_a	double	$\pi/6.0$ radians	A rotation movement needs to be performed before performing the filter update.
resample_interval	int	2	The number of filter updates required before resampling.

Parameter	Type	Default	Description
transform_tolerance	double	0	The time at which the published transformation will be post-processed to indicate that the transformation will be effective in the future.
recovery_alpha_slow	double	0	The exponential decay rate of the slow average weight filter is used to decide when to recover by adding random poses. A good value may be 0.001.
recovery_alpha_fast	double	0.0m	The exponential decay rate of the fast average weight filter is used to decide when to recover by adding random poses. A good value may be 0.1.
initial_pose_x	double	0.0m	The initial pose average (x), used to initialize the filter with Gaussian distribution.
initial_pose_y	double	0.0rad	The initial pose average (y), used to initialize the filter with Gaussian distribution.
initial_pose_a	double	0.5 *	The initial pose average (yaw), used to

Parameter	Type	Default	Description
		0.5m	initialize the filter with Gaussian distribution.
initial_cov_xx	double	0.5 * 0.5m	The initial pose covariance (x * x), used to initialize the filter with Gaussian distribution.
initial_cov_yy	double	-1.0 Hz	The initial pose covariance (y * y), used to initialize the filter with Gaussian distribution.
initial_cov_aa	double	0.5 Hz	The initial pose covariance (yaw * yaw), used to initialize the filter with Gaussian distribution.
gui_publish_rate	double	FALSE	The maximum rate (Hz) of publishing visual scans and paths. -1.0 is disabled.
save_pose_rate	double	FALSE	Store the maximum rate (Hz) of the last estimated pose and covariance of the parameter server in the variables ~initial_pose_ and ~initial_cov_. This saved pose will be used in subsequent runs to initialize the filter. -1.0 is

Parameter	Type	Default	Description
			disabled.
use_map_topic	bool	-1	When set to be true, AMCL will subscribe to the map topic instead of making a service call to receive its map.
first_map_only	bool	-1	When set to be true, AMCL will only use the first mapping it subscribes to instead of updating each time a new mapping is received.

#### 7.4 Configurable parameters in DWA

Parameter	Type	Default	Description
acc_lim_x	double	2.5	Robot's x acceleration limit (m/s <sup>2</sup> )
acc_lim_y	double	2.5	Robot's y acceleration limit (m/s <sup>2</sup> )
acc_lim_th	double	3.2	Robot's rotational acceleration limit (m/s <sup>2</sup> )
max_vel_trans	double	0.55	The absolute value of the maximum translational velocity of the robot (m/s).
min_vel_trans	double	0.1	The absolute value of the minimum



Parameter	Type	Default	Description
			translational velocity of the robot (m/s).
max_vel_x	double	0.55	Robot's maximum x velocity (m/s)
min_vel_x	double	0.0	Robot's minimum x velocity (m/s), negative when moving in reverse
max_vel_y	double	0.1	Robot's maximum y velocity (m/s)
min_vel_y	double	-0.1	Robot's minimum y velocity (m/s)
max_rot_vel	double	1.0	The absolute value of the maximum rotation velocity of the robot (rad/s)
min_rot_vel	double	0.4	The absolute value of the minimum rotation velocity of the robot (rad/s)
yaw_goal_tolerance	double	0.05	The radian tolerance of the yaw/rotation when the controller achieves its goal
xy_goal_tolerance	double	0.10	The tolerance of the controller in the distance between x and y when achieving the goal (m/s)
latch_xy_goal_tolerance	bool	false	If the goal tolerance is locked, when

Parameter	Type	Default	Description
			the robot reaches the goal xy position, it will simply rotate into position, even if it eventually exceeds the goal tolerance while doing so.
sim_time	double	1.7	Time to simulate the trajectory forward in seconds
sim_granularity	double	0.025	Step taken between points on a given trajectory (m/s)
vx_samples	int	3	The number of samples used when exploring the x velocity space
vy_samples	int	10	The number of samples used when exploring the y velocity space
vth_samples	int	20	The number of samples used when exploring the theta velocity space
controller_frequency	double	20.0	Call the controller's frequency. If it is not set in the controller's namespace, use searchParam to read the parameters from the parent namespace. Use together with

Parameter	Type	Default	Description
			move_base, which means you only need to set its "controller_frequency" parameter and you can safely not set this parameter.
path_distance_bias	double	32.0	The weight that how close the controller should be to the given path
goal_distance_bias	double	24.0	The weight that the controller should try to reach its local goal and it should also control the velocity
occdist_scale	double	0.01	The weight that the controller should try to avoid obstacles
forward_point_distance	double	0.325	The distance from the center of the robot to the additional scoring point, in meters
stop_time_buffer	double	0.2	The amount of time the robot must stop before colliding for the trajectory to be valid, in seconds
scaling_speed	double	0.25	The absolute value of the speed at which the robot's footprint is scaled

Parameter	Type	Default	Description
			(m/s)
max_scaling_factor	double	0.2	The biggest factor in scaling a robot's footprint
publish_cost_grid	bool	false	Whether the cost grid that the planner will use when planning will be published? When it's true, sensor_msgs/PointCloud2 will be available on the ~/cost_cloud topic. Each point cloud represents a cost grid and has a field for each individual scoring function component and the total cost of each cell, taking the scoring parameters into account.
oscillation_reset_dist	double	0.05	How far the robot must move in meters before resetting the oscillation tag
prune_plan	bool	true	Define whether the robot will eat the plan when moving along the path. If it's set to be true, the points will fall

Parameter	Type	Default	Description
			from the end of the plan as soon as the robots move more than 1 meter.

# Generation ROBOTS

Brand of the group **NGX** ROBOTICS

## **Official Distributor**

[gr@generationrobots.com](mailto:gr@generationrobots.com)

+33 5 56 39 37 05

[www.generationrobots.com](http://www.generationrobots.com)

