SCOUT MINI

Research and Development Kit and Pro

User Manual V.2.0.0 2021.04

This chapter contains important safety information, before the robot is powered on for the first time, any individual or organization must read and understand this information before using the device. If you have any questions about use, please contact us at support@agilex.ai. Please follow and implement all assembly instructions and guidelines in the chapters of this manual, which is very important. Particular attention should be paid to the text related to the warning signs.

⚠ Safety Information

The information in this manual does not include the design, installation and operation of a complete robot application, nor does it include all peripheral equipment that may affect the safety of the complete system. The design and use of the complete system need to comply with the safety requirements established in the standards and regulations of the country where the robot is installed. SCOUT MINI R&D kit integrators and end customers have the responsibility to ensure compliance with the applicable laws and regulations of relevant countries, and to ensure that there are no major dangers in the complete robot application. This includes but is not limited to the following:

1.Effectiveness and responsibility

- Make a risk assessment of the complete robot system.
- Connect the additional safety equipment of other machinery defined by the risk assessment together.
- Confirm that the design and installation of the entire robot system's peripheral equipment, including software and hardware systems, are correct.
- This robot including but not limited to automatic anti-collision, anti-falling, biological approach warning and other related safety functions. Related functions require integrators and end customers to follow relevant regulations and feasible laws and regulations for safety assessment , To ensure that the developed robot does not have any major hazards and safety hazards in actual applications.
- Collect all the documents in the technical file: including risk assessment and this manual.
- Know the possible safety risks before operating and using the equipment.

2.Environmental Considerations

- For the first use,please read this manual carefully to understand the basic operating content and operating specification.
- Use SCOUT MINI R&D kit always under 0°C~40°C ambient temperature.

3.Pre-work Checklist

- Make sure each device has sufficient power.
- Make sure SCOUT MINI R&D kit does not have any obvious defects.
- Check if the remote controller battery has sufficient power.
- When using, make sure the emergency stop switch has been released.

4.Operation

- In remote control operation, make sure the area around is relatively spacious.
- Carry out remote control within the range of visibility.
- When installing an external extension on SCOUT MINI R&D kit , confirm the position of the center of mass of the extension and make sure it is at the center of rotation.
- When SCOUT MINI R&D kit has a defect, please immediately stop using it to avoid secondary damage.
- When SCOUT MINI R&D kit has had a defect, please contact the relevant technical to deal with it, do not handle the defect by yourself.
- Always use SCOUT MINI R&D kit in the environment with the protection level requires for the equipment.
- Do not push SCOUT MINI R&D kit directly.
- When charging, make sure the ambient temperature is above 0°C.

5.Maintenance

- If the tire is severely worn or burst, please replace it in time.
- If the battery do not use for a long time, it need to charge the battery periodically in 2 to 3 months.

CONTENTS

0 Product Overview

SCOUT MINI R&D Kit

SCOUT MINI R&D Kit Pro

SCOUT MINI Research&Development Kit & Pro is an entry-level version and advanced set of ROS developers developed and customized by Agliex Robotics for ROS scientific research and education applications. Integrated with high-performance industrial control, high-precision LiDAR and multiple sensors based on the Agliex Robot ROS ecosystem, which is capable to achieve applications such as mobile robot motion control, communication, navigation, map building, etc. It provides with complete developer documentation and DEMO resources, lightweight and portable, highly technological industrial design, and exclusive sensor customized holder. The best experimental platform for rapid secondary development of ROS for multi-directional applications such as education and scientific research, product pre-research, subjects and product demonstration.

1 Configuration list and Parameters

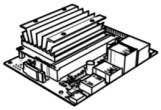
1.1 SCOUT MINI R&D Kit & Pro Comparison

Version	SCOUT MINI R&D Kit	SCOUT MINI R&D Kit Pro
Mobile base	SCOUT MINI Off-road	SCOUT MINI Off-road
Computing	Nvidia Jetson Nano	Nvidia AGX Xavier
Lidar	EAI-G4	VLP -16
Camera	Intel RealSense D435	Intel RealSense D435
Communication	GL.iNet GL-AR750S	GL.iNet GL-AR750S

1.2 Shipping List

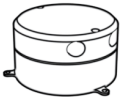
1.2.1 SCOUT MINI R&D Kit

Name	Quality	Model
Computing Unit	1	Nvidia Nano 4G
Laser Sensor	1	EAI-G4
Vision Sensor	1	Intel RealSense D435
Router	1	GL.iNet GL-AR750S
Mobile base	1	SCOUT MINI
Remote Control	1	FS-I6S
Charger	1	Agilex UY360
Voltage Stabilizer	1	24V To12V
Voltage Stabilizer	1	12V To 5V
HD Display	1	
USB To CAN	1	
USB HUB	1	



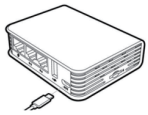
Nvidia Jetson Nano

Intel RealSense D435



EAI G4

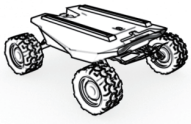
USB-HUB



Router

HD Display

USB To CAN

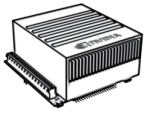


SCOUT MINI

SCOUT MINI R&D Kit Configuration

1.2.2 SCOUT MINI R&D Kit Pro

Name	Quality	Model
Computing Unit	1	Nvidia AGX Xavier
Laser Sensor	1	VLP -16
Vision Sensor	1	Intel RealSense D435
Router	1	GL.iNet GL-AR750S
Mobile base	1	SCOUT MINI
Remote Control	1	FS-I6S
Charger	1	Agilex UY360
Voltage Stabilizer	1	24V To 19 V
Voltage Stabilizer	1	24V To 12V
HD Display	1	
USB To CAN	1	
USB HUB	1	

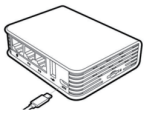


Nvidia AGX Xavier

Intel RealSense D435

VLP 16

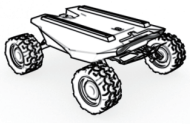
USB-HUB



Router

HD Display

USB To CAN



SCOUT MINI

SCOUT MINI R&D Kit Pro Configuration

1.3 SCOUT MINI Introduction

1.3.1 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". SCOUT MINI inherits the advantages of the SCOUT four-wheel differential chassis series with 4WD, independent suspension, no turning radius and has made innovations in the design of the hub motor. The minimum turning radius of the chassis is 0m, and the climbing angle is close to 30 degrees. SCOUT MINI is half 50 % smaller than SCOUT, while still having excellent off-road performance. At the same time, it has achieved a 10.8km/h high-speed, precise, stable and controllable power control system. The SCOUT MINI development platform has its own control system, supports standard CAN protocal and connected to various external devices. Furthermore, it supports ROS/Autoware secondary development and advanced robotics development. Accessories included standard activation plug, 24V@15Ah lithium battery and endurance mileage up to 10KM.

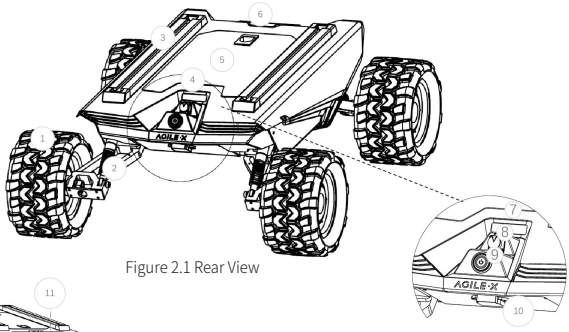


Figure 2.1 Rear View

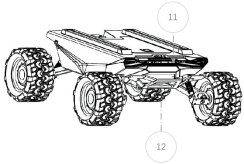


Figure 2.1 Front View

1. Tire
2. Spring shock absorber
3. Extension support
4. Control interface area
5. Electrical bin panel
6. Tail light
7. Power capacity LCD
8. CAN extension interface
9. Power key
10. Charging interface
11. Font light
12. Front fence

1.3.2 SCOUT MINI Instruction

1.3.2.1 SCOUT MINI Checking

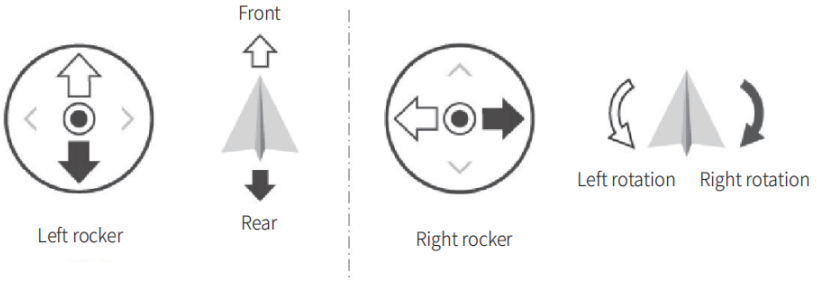
- Press the power button of and wait for a few seconds
- Charge the battery if the SOC is less than 30%
- Please use the standard charger and power off the system while charging
- It takes about one and half hour to fully charged the battery

1.3.2.2 Remote Control

- Check the remote control battery
- Put SWA、SWB、SWC、SWD all to up positions
- Hold the power button 1 and 2 at the same timeuntil turned on

1.3.2.3 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 rocker controls the Scout mini move forward and backward while the right rocker 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 control the control mode, top position is the command mode, remote control mode is the middle position, and the bottom position the 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 control mode, other modes are invalid.
- 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, other modes are invalid.

1.3.2.4 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 mobile base will stop;
- Try to gently push the right joystick to the left, it can see that the car slowly rotates to the left;
- Try to gently push the right joystick to the right, it can see that the car slowly rotates to the right;
- Release the right joystick and the mobile base will stop;
- Then may try to control the mobile base in the relatively open area and be familiar with the speed control of the vehicle。

1.3.2.5 SCOUT MINI Shut down

- Press SCOUT MINI power button and then release。

1.3.2.6 Remote Control Shut down

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

1.3.3 SCOUT MINI Parameters

SCOUT MINI Parameters Description	
Size	627mm 550mm 252mm
Wheelbase	452mm
Wheelbase	450mm
Weight	20kg
Minimum Ground Clearance	107mm
Standard Load	10kg(0.5 Coefficient of friction)
Maximum Speed	10.8km/h
Minimum Turning Radius	0(Self-turning)
Maximum Slope	30°(with load)
Obstacle Surmounting	70mm
Maximum mileage	10km(without load)
Drive 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

1.4 Nvidia Jetson Nano

Nvidia Jetson Nano is a powerful and compact computer which designed to support entry-level AI applications and devices. It contains multiple acceleration libraries for deep learning, computer vision, graphics and multimedia based on the comprehensive NVIDIA JetPack™ SDK. It is installed on the SCOUT MINI R&D Kit version and can be used to expand the applications of robot navigation and positioning, image processing, voice recognition and various other technologies.

GPU	128-Core Maxwell
CPU	Quad-core ARM57 @1.43Ghz
Memory	4GB 64Bit LPDDR4 25.6GB/s
Storage	Micro SD卡 (default)
Encoder	4K@30	4 X 1080p@30	9 X 720p@30(H.264/H.265)
Decoder	4K@60| 2X 4K@30	8X 1080p@30	18 X 720p@30(H.264/H.265)
Camera	2 X MIPI CSI-2 DPHY lanes
Internet	Gigabit Ethernet, M.2 Key E interface
Display	HDMI X 1, DP X 1
USB	4 X USB 3.0, USB 2.0 Micro-B
Extension Interface	GPIO, I2C, I2S, SPI, UART

1.5 Nvidia Xavier

The NVIDIA Jetson AGX XAVIER Developer Kit is capable to run modern AI workloads and solve problems in optical inspection, manufacturing, robotics, logistics, retail, service, agriculture, smart cities, and healthcare. Plus, it delivers up to 32 TOPS and can operate in as little as 10 W.

Jetson AGX Xavier Developer Kit is supported by NVIDIA JetPack, which includes a board support package (BSP), Linux OS, NVIDIA CUDA, cuDNN, and TensorRT software libraries for deep learning, computer vision, GPU computing, multimedia processing, and much more. It' s also supported by the NVIDIA DeepStream SDK, which delivers a complete toolkit for real-time situational awareness through intelligent video analytics (IVA) and NVIDIA Isaac SDK, which delivers a software toolkit for robot development. These helps boost performance and accelerate software development, while reducing development cost and effort.

Development Kit Technical Parameters	
GPU	Tensor Core 512 -Volta GPU
CPU	8-Codes ARM v8.2 64 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) Close-System Debug and Flashing Support on 1 Port
Camera Connector	(16*) CSI-2 Lanes
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 Header	High-Definition Audio
eSTATp + USB 3.0 Type A	SATA Through PCIe x1 Bridge (PD + Data for 2.5-inch SATA) + USB 3.0
HDMI Type A	HDMI 2.0
µSD/UFS	SD/UFS

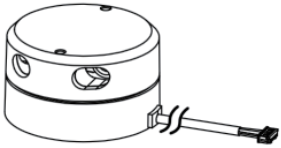
1.6 Intel RealSense D435

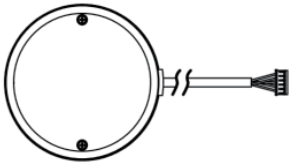
The Intel® RealSense™ depth camera D435 is a stereo solution, offering quality depth for a variety of applications. It's wide field of view is perfect for applications such as robotics or augmented and virtual reality, where seeing as much of the scene as possible is vitally important. With a range up to 10m, this small form factor camera can be integrated into any solution with ease, and comes complete with our Intel RealSense SDK 2.0 and cross-platform support.

	Item	Intel Realsense D435
Features	Use Environment	Indoor/Outdoor
	Range	Approximately 10m
	Image sensor technology	Global Shutter, 3um X 3um pixel size
	IMU Support	N/A
Depth Camera	Depth Technology	Active IR Stereo
	FOV	86° x 57° (±3°)
	Minimum Depth Distance	0.105m
	Depth Output Resolution	1280 x 720
	Maximum Measurement Distance	Approximately 10m
	Depth Frame Rate	90 fps
	RGB Frame Resolution	1280 x 800
RGB	FOV	69.4° × 42.5° (±3°)
	RGB Frame Rate	30fps
Others	Size	90mm x 25mm x 25mm
	Interface Type	USB-C 3.1

1.7 EAI-G4

YDLIDAR G4 is a 360-degree two-dimensional rangefinder (hereinafter referred to as G4) developed by YDLIDAR team. Based on the principle of triangulation, it is equipped with related optics,electricity, and algorithm design to achieve high-frequency and high-precision distance measurement. The mechanical structure rotates 360 degrees to continuously output the angle information as well as the point cloud data of the scanning environment while ranging.

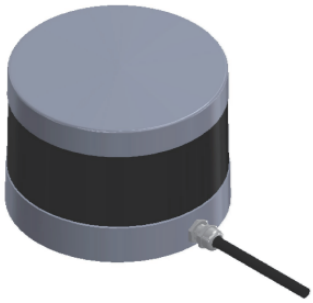




1.8 Velodyne VLP-16

Velodyne VLP-16 is the smallest, cost-optimized product in Velodyne' s 3D LiDAR product range. Developed with mass production in mind, the VLP-16 is far more cost-effective than comparable sensors, and it retains the key features of Velodyne' s breakthroughs in LiDAR:Real-time, 360°, 3D distance and calibrated reflectivity measurements.
The VLP-16 has a range of 100 m, and the sensor's low power consumption, light weight, compact footprint and dual return capability make it ideal not only for autonomous vehicles but also for robotics, terrestrial 3D mapping and many other applications.

Item	Parameters
Measurement Range	100m
Range Accuracy	±3cm
Scanning speed	Single way 300,000 points/s Round way 600,000 points/s
Field of View(Vertical)	-15°~+15°
Angular Resolution(Vertical)	2°
Rotational Rate	5Hz~20Hz
Laser Product Classification	Class 1
Weight	830g
Power consumption	8W
Operating Voltage	9V~18V
Operating Temperature	-10°C~+60°C

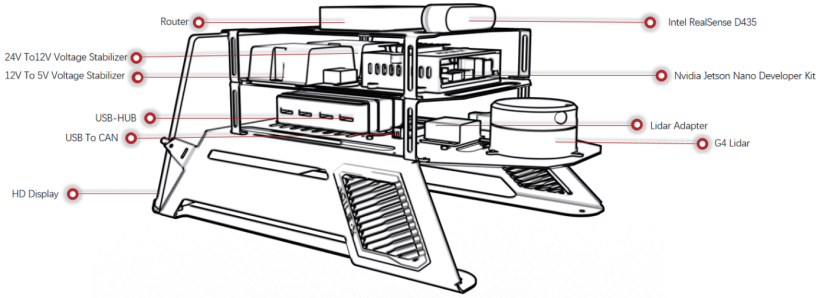


2 Hardware Installation and Electricity

2.1 SCOUT MINI R&D Kit & Pro Kit Installation

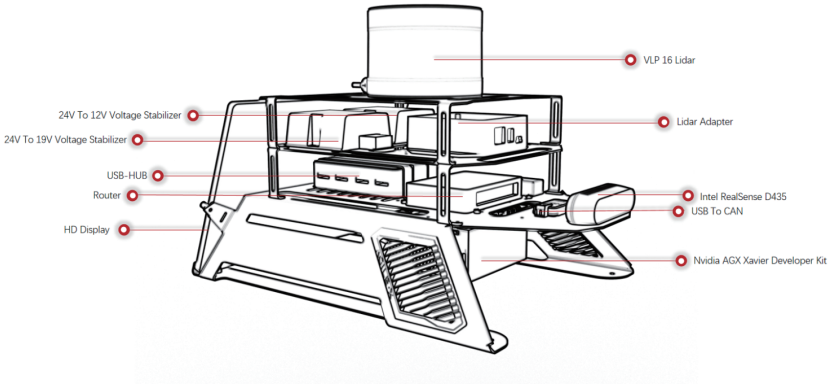
2.1.1 SCOUT MINI R&D Kit

The overall design of the SCOUT MINI R&D Kit research and education kit uses the stacking design and the frame is made of sheet metal, a high-definition display screen is installed at the rear part which is convenient for customers to develop and debug. A hollow sheet metal holder is used at the bottom to assemble with the standard aluminum holder of the SCOUT MINI above. The USB -HUB with independent power supply is located at the second layer along with USB to CAN module and G4 lidar. The third layer is mainly composed of computing units and voltage stabilizing modules and the top layer is currently mainly installed with Intel RealSense D435 . Please refer to the following figure for detail.



2.1.2 SCOUT MINI R&D Kit Pro

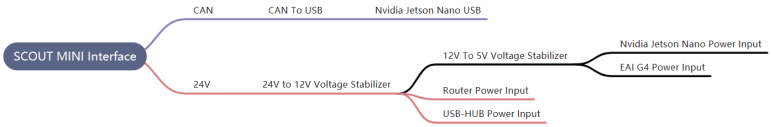
The SCOUT MINI R&D Kit share the same basic design with SCOUT MINI R&D Kit, the difference between each other are the sensors location. The computing unit Nvida AGX Xavier is at the bottom of the holder, the USB-HUB with independent power supply is located at the second layer along with USB to CAN module and Intel RealSense D435 . The third layer is mainly composed of lidar adapter and voltage stabilizing modules, while the top layer is currently mainly installed with VLP 16 lidar . Please refer to the following figure for detail.



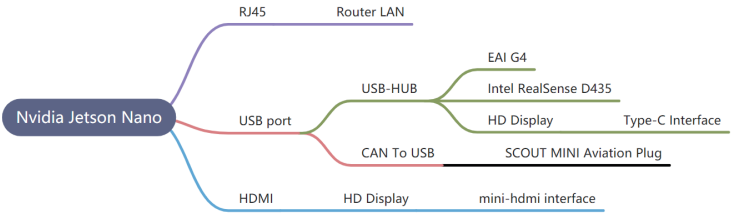
2.2 Electricity and Communication Connection

2.2.1 SCOUT MINI R&D Kit

SCOUT MINI R&D Kit provides power and communication interfaces for devices and sensors through the aviation expansion interface of the chassis. The SCOUT MINI chassis power expansion interface (maximum power output supports 24V@5A) is powered by the chassis' battery, without a voltage regulator and voltage regulation module. In order to provide power supply to devices and sensors, the voltage regulator and stabilizer is essential. We chose a 12V@20A voltage stabilizer module. The voltage stabilizer mainly provides power input for 5V voltage stabilizer modules, wireless routers, and USB-HUB. The 5@10A voltage stabilizer module provide power input for EAI-G4 laser Radar and Nvidia Jetson Nano.

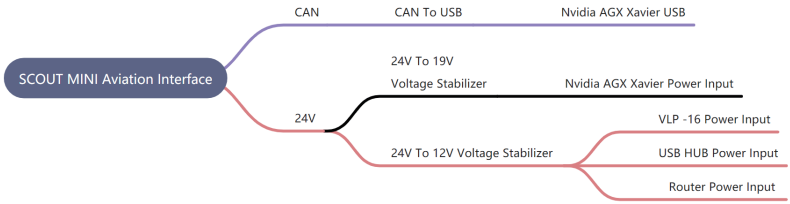


The following figure is the external connection of Nvidia Jetson Nano. Nano's network port is connected to the router's network port which is designed for remote desktop control to access and debug, also it is easy to expand other network devices. For the USB expansion, we add a USB-HUB (with additional power supply) to improve the devices expansion capacity. USB-HUB mainly connect with EAI-G4 lidar, Intel RealSenseD435,HD display, etc.

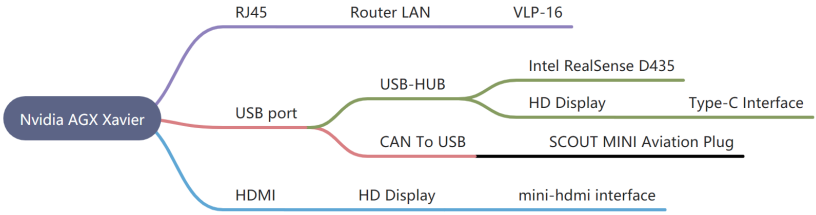


2.2.2 SCOUT MINI R&D Kit Pro

The following figure is the external connection of Nvidia Jetson Nano. Nano's network port is connected to the router's network port which is designed for remote desktop control to access and debug, also it is easy to expand other network devices. For the USB expansion, we add a USB-HUB (with additional power supply) to improve the devices expansion capacity. USB-HUB mainly connect with EAI-G4 lidar, Intel RealSenseD435,HD display, etc.。



The following figure is the external connection of Nvidia Jetson Nano. Nano's network port is connected to the router's network port which is designed for remote desktop control to access and debug, also it is easy to expand other network devices. For the USB expansion, we add a USB-HUB (with additional power supply) to improve the devices expansion capacity. USB-HUB mainly connect with Intel RealSenseD435,HD display, etc.。



2.3 Sensors expansion

External expansion mainly involves mechanical installation expansion, power supply expansion, and communication expansion. For the power supply expansion, we took this issue into consideration when selecting the power supply voltage regulator in the early stage, so the power supply voltage regulator module has a certain margin. For communication expansion, USB - HUB and wireless gateway are added to the equipment. So it is capable to access more devices.

3 Development Guide

3.1 ROS Development Introduction

3.1.1 ROS History

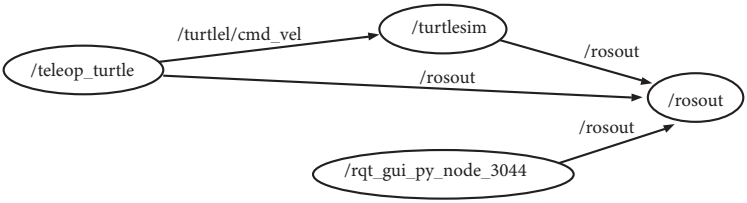
Robot Operating System (ROS or ros) is an open source robotics middleware suite. Although ROS is not an operating system but a collection of software frameworks for robot software development, it provides services designed for a heterogeneous computer cluster such as hardware abstraction, low-level device control, implementation of commonly used functionality, message-passing between processes, and package management. Running sets of ROS-based processes are represented in a graph architecture where processing takes place in nodes that may receive, post and multiplex sensor data, control, state, planning, actuator, and other messages. Despite the importance of reactivity and low latency in robot control, ROS itself is not a real-time OS (RTOS). It is possible, however, to integrate ROS with real-time code. The lack of support for real-time systems has been addressed in the creation of ROS 2.0,a major revision of the ROS API which will take advantage of modern libraries and technologies for core ROS functionality and add support for real-time code and embedded hardware.

3.1.2 ROS Concept

ROS has three levels of concepts: the Filesystem level, the Computation Graph level, and the Community level. These levels and concepts are summarized below and later sections go into each of these in greater detail.

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

ROS processes are represented as nodes in a graph structure, connected by edges called topics. ROS nodes can pass messages to one another through topics, make service calls to other nodes, provide a service for other nodes, or set or retrieve shared data from a communal database called the parameter server. A process called the ROS Master[66] makes all of this possible by registering nodes to itself, setting up node-to-node communication for topics, and controlling parameter server updates. Messages and service calls do not pass through the master, rather the master sets up peer-to-peer communication between all node processes after they register themselves with the master. This decentralized architecture lends itself well to robots, which often consist of a subset of networked computer hardware, and may communicate with off-board computers for heavy computation or commands.

3.1.4.1 ROS Topic

Topics are named buses over which nodes send and receive messages. Topic names must be unique within their namespace as well. To send messages to a topic, a node must publish to said topic, while to receive messages it must subscribe. The publish/subscribe model is anonymous: no node knows which nodes are sending or receiving on a topic, only that it is sending/receiving on that topic. The types of messages passed on a topic vary widely and can be user-defined. The content of these messages can be sensor data, motor control commands, state information, actuator commands, or anything else.

3.1.4.2 ROS Service

A node may also advertise services. A service represents an action that a node can take which will have a single result. As such, services are often used for actions which have a defined beginning and end, such as capturing a single-frame image, rather than processing velocity commands to a wheel motor or odometer data from a wheel encoder. Nodes advertise services and call services from one another.

3.1.4.3 ROS File System

Similar to an operating system, ROS files are also organized on the hard disk in a particular fashion.
Package：The ROS packages are the most basic unit of the ROS software. It contains the ROS runtime process (nodes), libraries, configuration files, and so on, which are organized together as a single unit. Packages are the atomic build item and release item in the ROS software.
Package Manifest：The package manifest file is inside a package that contains information about the package, author, license, dependencies, compilation flags, and so on. The package.xml file inside the ROS package is the manifest file of that package.
Meta Package：The term meta package is used for a group of packages for a special purpose. In an older version of ROS such as Electric and Fuerte, it was called stacks, but later it was removed, as simplicity and meta packages came to existence. One of the examples of a meta package is the ROS navigation stack.

3.2 Start up and shut down

3.2.1 Installation

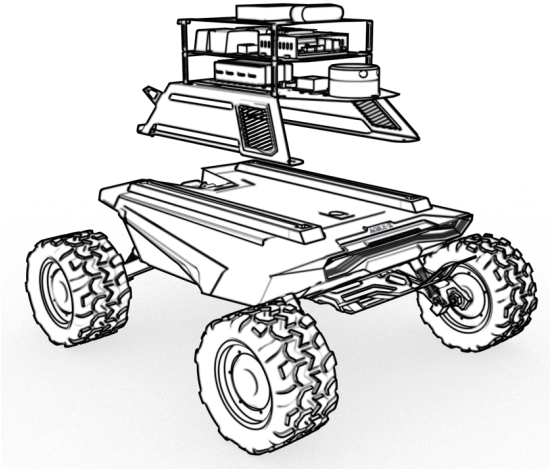
3.2.1.1 Tool

Hexagon key

3.2.1.2 Sensors holder and mobile base

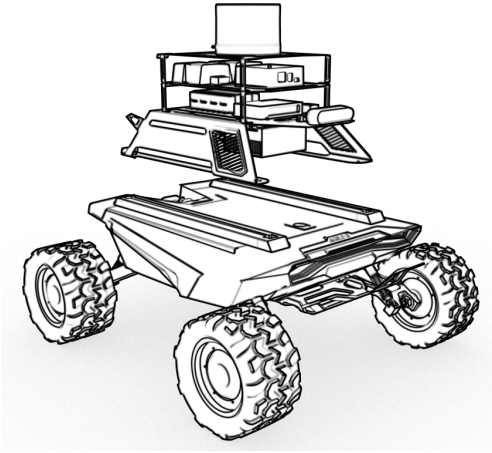
SCOUT MINI R&D Kit

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.



SCOUT MINI R&D Kit Pro

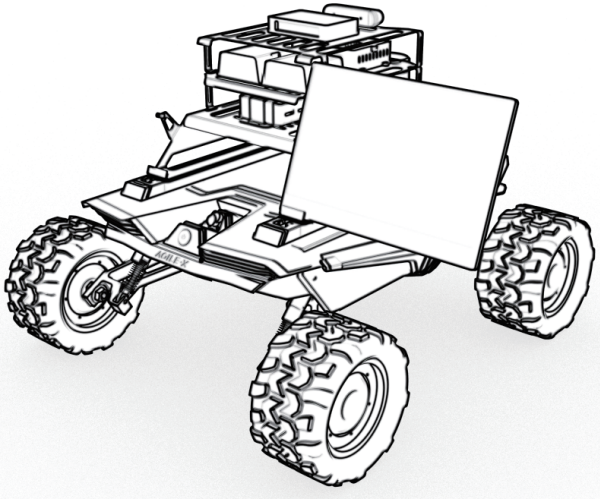
When the product is ready for shipping, the sensor holder is separated from the Scout MINI. User need to use tool to set the sensor holder on the Scout MINI. Firstly, put the four slider nuts into the slide way each side on the Scout MINI , and then use the hexagonal tool to screw corresponding four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



3.1.1.3 HD Display Installation

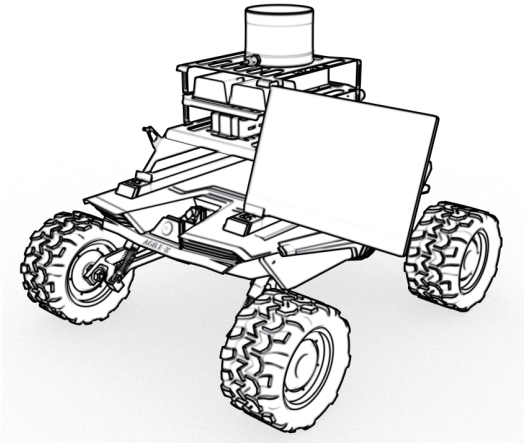
SCOUT MINI R&D Kit

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..



SCOUT MINI R&D Kit 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 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.

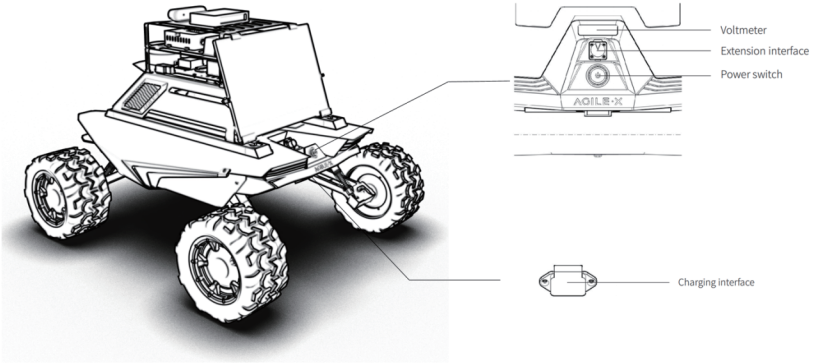
3.2.3 Mouse and keyboard

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

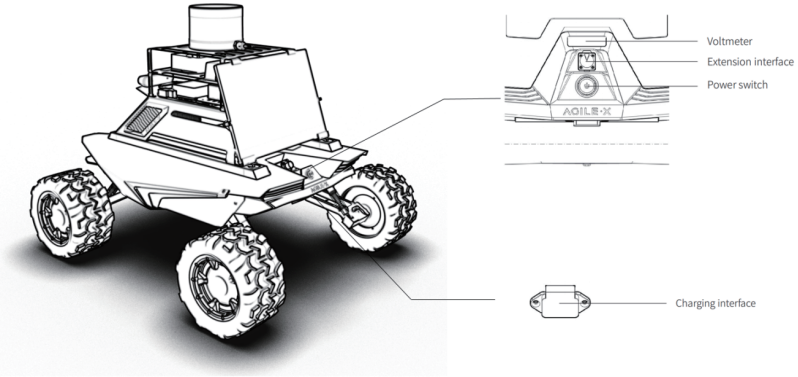
3.2.4.1 SCOUT MINI R&D Kit

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



3.2.4.2 SCOUT MINI R&D Kit 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 he system, do not press the power button directly because thel 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.

3.2.6 Shut Down

If you need to shut down he system, do not press the power button directly because thel 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 R&D Kit & Pro Development Environment

(Ubuntu, ROS, Gazebo, RVIZ)

Ubuntu	16.04	Ubuntu	18.04
ROS	kinetic	ROS	melodic
Gazebo	7.0	Gazebo	9.0
RVIZ	-	RVIZ	-

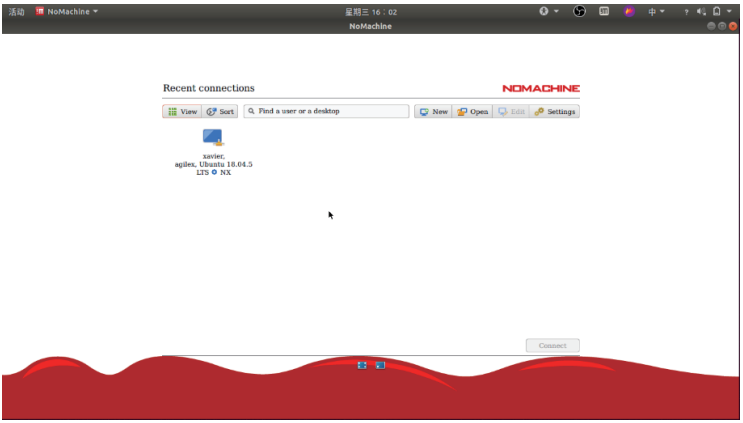
3.4 Remote Desktop

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

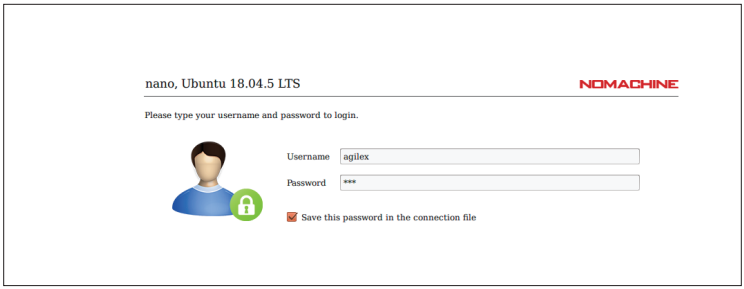
3.4.2 Xavier or Nano Connection(make sure the computer unit is power on)

3.4.2.1 Choose connection object



3.4.2.2 Click Yes

3.4.2.3 Username：agilex Password：agx choose save password

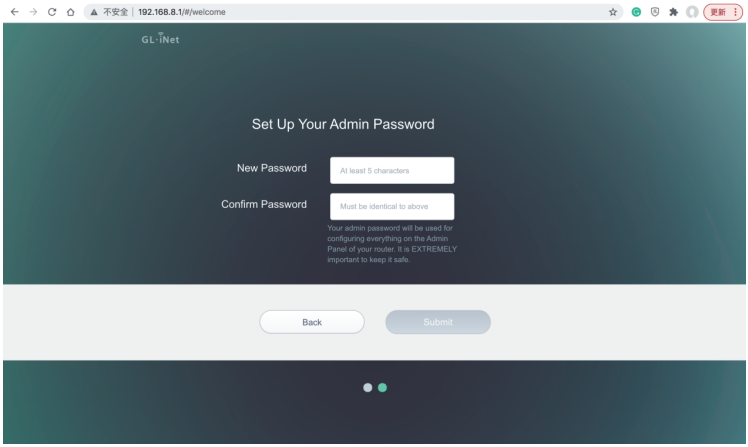


3.4.2.4 keep clicking OK

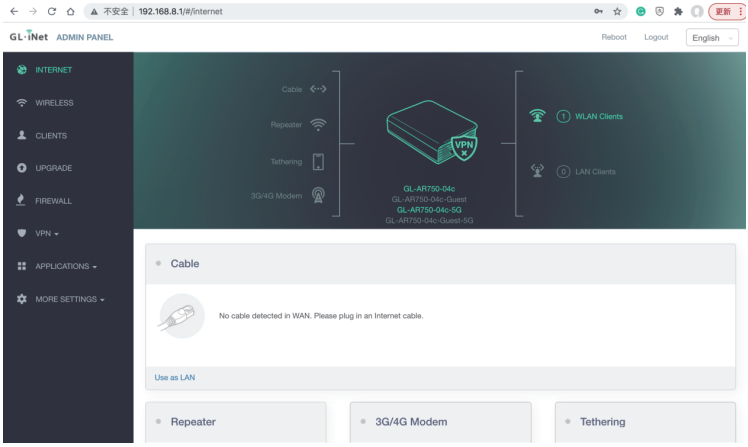
3.4.2.5 Enter default password agx

3.4.2.6 Connection successful

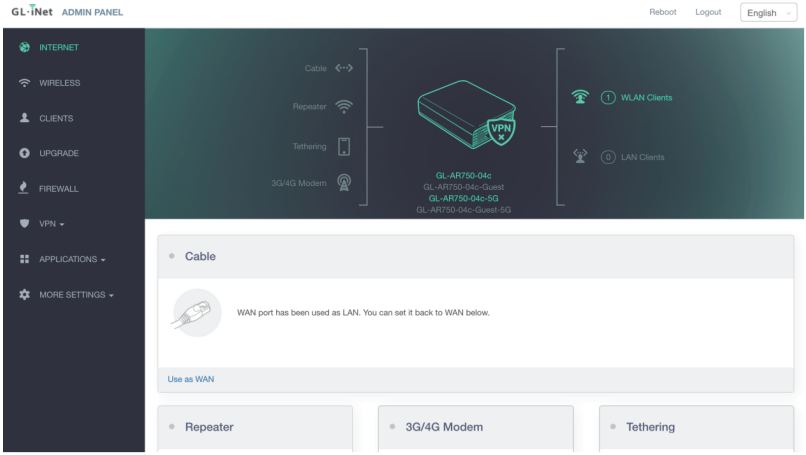
3.4.3.1 Enter the router, the default address is 192.168.8.1 select the language and set the password



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 the 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 C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

Update

```
sudo apt-get update
```

ROS Desktop-Full Ubuntu16.04

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

Ubuntu18.04

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

Dependency

```
sudo rosdep in
rosdep update
```

Environment set up
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，

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

Check CAN

```
ifconfig -a
```

Testing the hardware by using can-utils

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

Command testing

```
# receiving data from can0
$ candump can0
# send data to can0
$ cansend can0 001#1122334455667788
```

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。

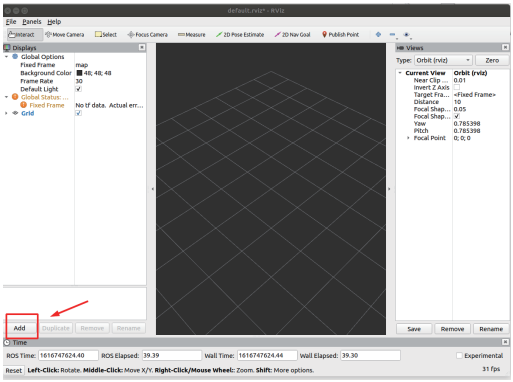
3.6.3 RealSense D435 node ROS Package and RVIZ visualization

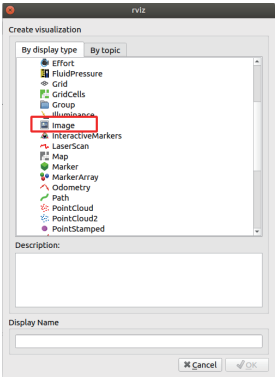
realsense2_camera: Set up and start the depth camera function package, and you can use the rviz visualization tool to view color maps, depth maps, dense point clouds, etc.

Get the rgb

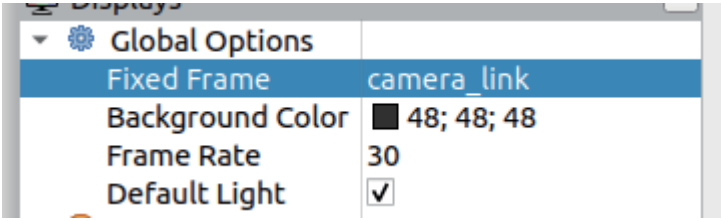
```
roslaunch realsense2_camera rs_camera.launch
```

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

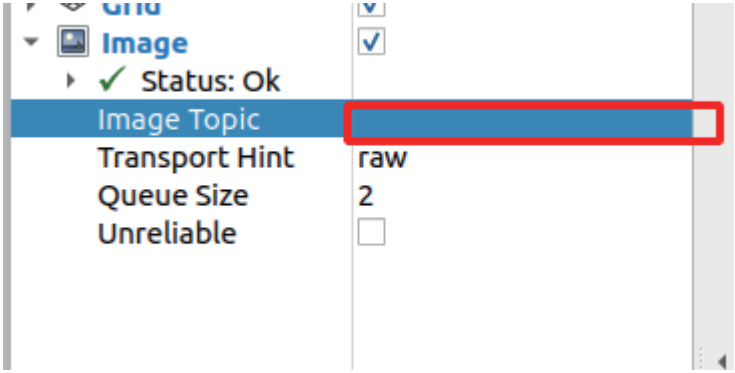


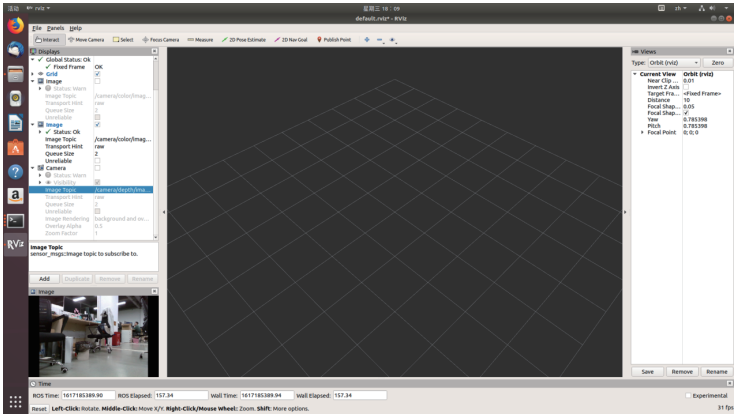


Fixed frame choose camera_link



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

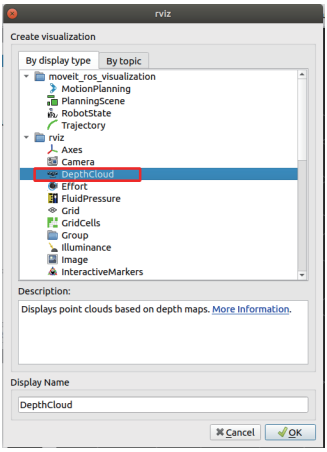




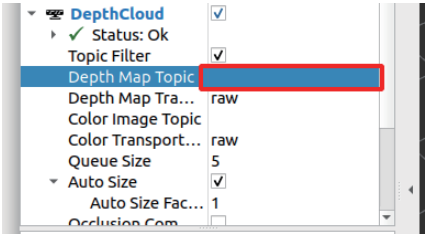
Get the rgb+Depth map

```
roslaunch realsense2_camera rs_camera.launch
```

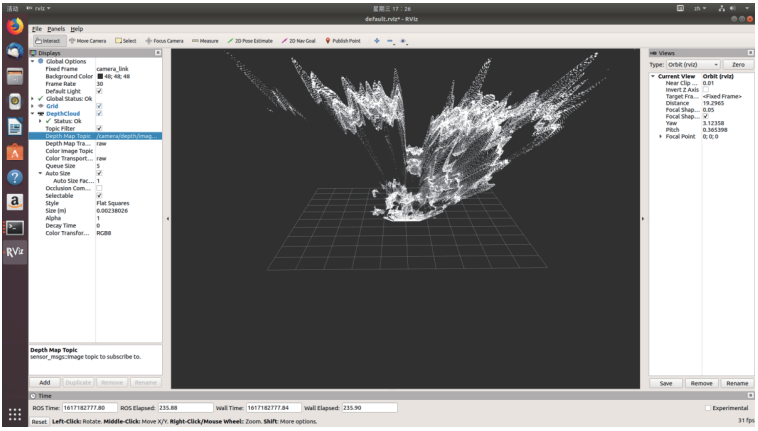
Open rviz, click the add to add DepthCloud component



Fixed frame choose camera_link, choose the corresponding topic in the Depth Map topic.



The depth map in shown below.



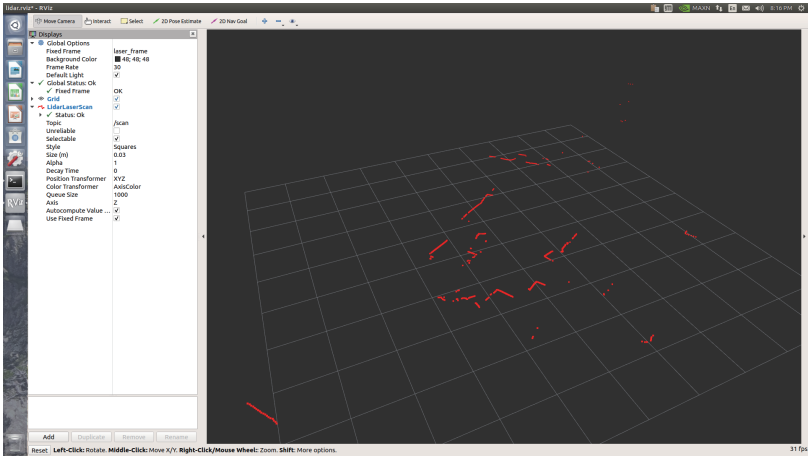
3.6.4 EAI -G4 ROS Package

ydlidar_ros: Start the EAI-G4 single-line lidar function package, and users can set some parameters of the radar according to specific application scenarios.

Show the scan result of the lidar

```
cd ydlidar/launch
```

```
roslaunch ydlidar lidar_view.launch
```



Start lidar, publish base_link-><laser_link> coordinate transfer

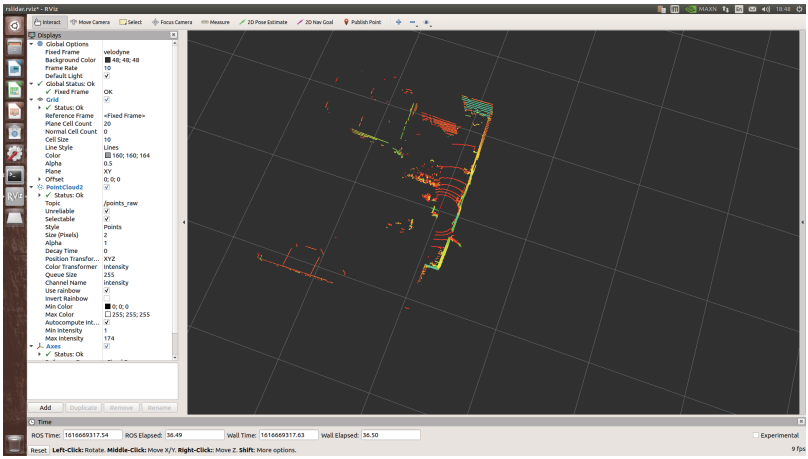
```
roslaunch scout_bringup open_rslidar.launch
```

3.6.5 VLP -16 ROS Package

velodyne_driver: Processing raw point cloud data and laser data function package
velodyne_laserscan: base on velodyne_driver package，Compress point cloud data，send /scan data
velodyne_pointcloud: Start all the nodes of the VLP_16 lidar and start publishing various data.
velodyne_msgs: Define various data formats of VLP_16 lidar。

Start lidar, publish base_link-><laser_link> coordinate transfer

```
roslaunch scout_bringup open_rslidar.launch
```



3.7 Navigation and positioning based on Gmapping open source architecture

3.7.1 2D Lidar

3.7.1.1 Navigation algorithm

Move_base is the core algorithm of robot navigation path planning. It subscribes to the positioning data of lidar /scan, map, /odom mileage and amcl, and then uses the path planning plugin integrated into move_base. The global path planning plugin includes:

- navfn: The older code version in ROS implements the dijkstra and A* global planning algorithms.
- global_planner: Reimplemented Dijkstra and A* global planning algorithm, the improved version of navfn.
- parrot_planner: A simple algorithm to achieve global path planning algorithm.
  Partial path planning plugins include：
- base_local_planner: Implemented two local planning algorithms, Trajectory Rollout and DWA.
- dwa_local_planner: The DWA local planning algorithm is implemented which can be regarded as an improved version of base_local_planner.

3.7.1.2 2D Lidar map building

ROS SLAM Algorithm：

- gmapping：Lidar/scan data and odometer/odom data are required, using PF (Particle Filter).
- hector ：Based on the optimized algorithm (solving the least squares problem), the advantages and disadvantages: no odometer is required, but the radar frame rate requires a very high rate at 40Hz, and the estimated 6-degree-of-freedom pose can adapt to the situation of uneven air or ground. Initial value has a great influence on the result, so the radar frame rate is required to be higher.
- Cartographer：The cumulative error is lower than the previous two algorithms, and it can naturally output the covariance matrix and the input term optimized by the back-end. Lower-cost radars can also have high performance.

  Combining these several slam algorithms, the kit uses gmapping algorithm to create slam maps. The slam_gmapping node receives the sensor_msgs/LaserScan message and builds a map (nav_msgs/OccupancyGrid). The map can be viewed through ROS topic or service.

Gmapping topics and service

	Name	Type	Description
Topics subscribe	tf	tfMessage	coordinate system, datum and distance measurement related frame conversion
	scan	sensor_msgs/LaserScan	Lidar scanning data
Topic publish	map_metadata	nav_msgs / MapMetaData	Publish Meta data
	map	nav_msgs / OccupancyGrid	Publish map raster data
	entropy	std_msgs / Float64	Publish the estimation of robot pose distribution entropy
Service	dynamic_map	nav_msgs / GetMap	Call this service to get map data

Gmapping Coordinate transformation

	TF transformation	Description
Necessary TF transformation	base_link-><laser_link>	Direct conversion between chassis coordinate system and lidar coordinate system
	odom->base_link	The transformation between the mileage coordinate system and the chassis coordinate system, generally issued by the odometer
Necessary TF transformation	map->odom	The transformation between the map coordinate system and the odometer coordinate system to estimate the pose of the robot in the map

3.7.1.3 2D Autonomous Navigation

SLAM Mapping:

(1) Start the lidar and publish the coordinate transformation of base_link-><laser_link>

```
roslaunch scout_bringup open_rslidar.launch
```

(2) Start gmapping mapping node

```
roslaunch scout_bringup gmapping.launch
```

(3) Use the remote control to control the Kit to move around the scene. After building the map, save the map to the specified directory (usually /description)

```
rosrun map_server map_saver –f ~/catkin_ws/scout _base/scout_description/maps/map
```

Autonomous Navigation:

(1) Start the lidar and publish the coordinate transformation of base_link-><laser_link>

```
roslaunch scout_bringup open_rslidar.launch
```

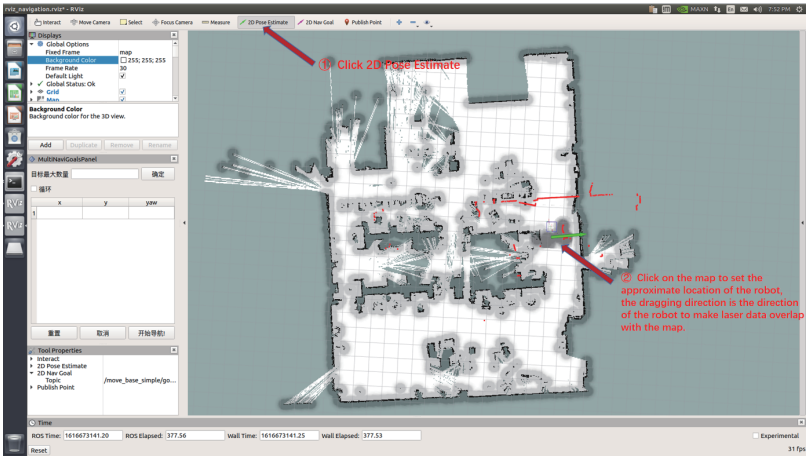
(2) Start move_base navigation

```
roslaunch scout_bringup navigation.launch
```

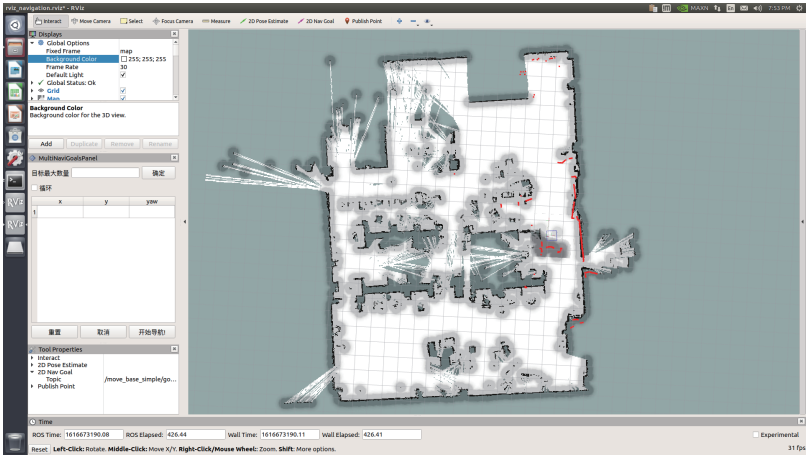
Note: If you need to choose different map, please open the navigation.launch file to modify the parameters which is marked with red line as shown in the figure below.

```
<?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">
    <rosparam file="$(find scout_description)/param/4wd/costmap_common_params.yaml" command="load" ns="global_costmap" />

    <rosparam file="$(find scout_description)/param/4wd/costmap_common_params.yaml" command="load" ns="local_costmap" />

    <rosparam file="$(find scout_description)/param/4wd/local_costmap_params.yaml" command="load" />

    <rosparam file="$(find scout_description)/param/4wd/global_costmap_params.yaml" command="load" />

    <rosparam file="$(find scout_description)/param/4wd/base_local_planner_params.yaml" command="load" />

    <rosparam 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">
            <rosparam 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>
```

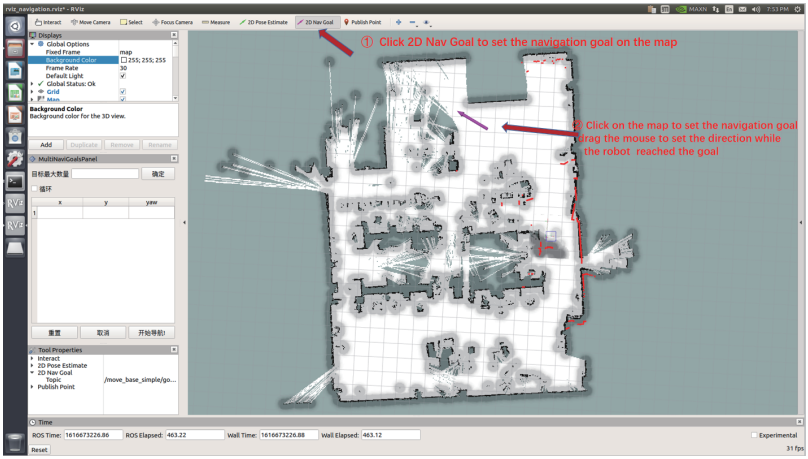
(3) Regulate the actual position of the robot on the map displayed in rviz, and regulate the chassis rotation by publishing an approximate position with the handle. When the laser shape overlaps with the scene shape in the map, the regulation is complete.



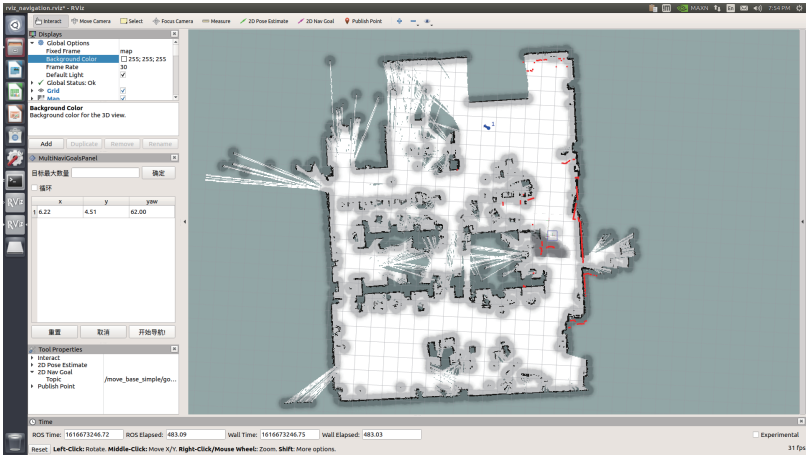
After setting the positioning, the laser shape and the scene shape in the map have basically overlapped, the means the regulation is complete. As the picture shown below.



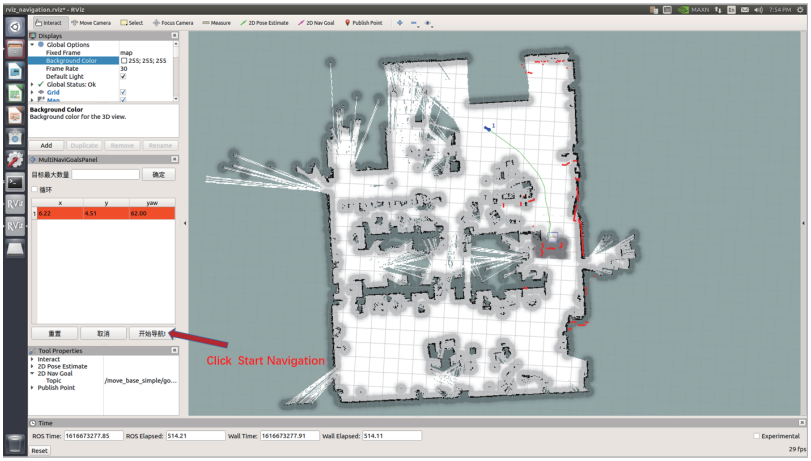
(4) Set multiple target points on the multi-point navigation plug-in in the left column, click Start Navigation, and switch the SWB to command control mode.



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



3.7.2 3D Lidar

3.7.2.1 3D Lidar map building

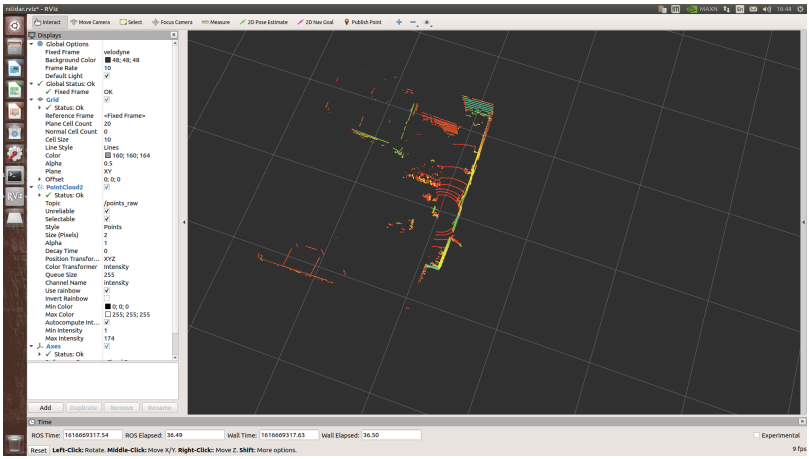
Take the VLP-16 radar as an example. The VLP-16 radar is a 16-line lidar. In 2° increments, 16° up and down respectively. Since the VLP directly outputs point cloud data, 16 lines of laser data are included. Gmapping mapping requires only one line of data, so point cloud data must be compressed. For this time, you can use the ros package provided by VLP to compress the point cloud data and output /scan. You can also use the pointcloud_to_laserscan package we provide to compress point cloud data. Since the 3D lidar has a sweep angle, the sweep blind area of the 3D radar will be less than that of the 2D radar. After obtaining the scan data, you can transfer the data to gmapping, and then provide a relatively reliable odometer information and correct coordinate system transformation and gmapping can be created.

3.7.2.2 3D Lidar Autonomous Navigation

SLAM Mapping:

(1) Start the lidar and publish the coordinate transformation of base_link-><laser_link>

```
roslaunch scout_bringup open_rslidar.launch
```



(2) Start gmapping mapping node

```
roslaunch scout_bringup gmapping.launch
```

Note: If you need to choose different map, please open the navigation.launch file to modify the parameters which is marked with red line as shown in the figure below.

```
<?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">
    <rosparam file="${find scout_description)/param/4wd/costmap_common_params.yaml" command="load" ns="global_costmap" />

    <rosparam file="${find scout_description)/param/4wd/costmap_common_params.yaml" command="load" ns="local_costmap" />

    <rosparam file="${find scout_description)/param/4wd/local_costmap_params.yaml" command="load" />

    <rosparam file="${find scout_description)/param/4wd/global_costmap_params.yaml" command="load" />

    <rosparam file="${find scout_description)/param/4wd/base_local_planner_params.yaml" command="load" />

    <rosparam 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">
                <rosparam 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) Use the remote control to control the Kit to move around the scene. After building the map, save the map to the specified directory (usually /description)

```
rosrun map_server map_saver –f ~/catkin_ws/scout _base/scout_description/maps/map
```

Autonomous Navigation：

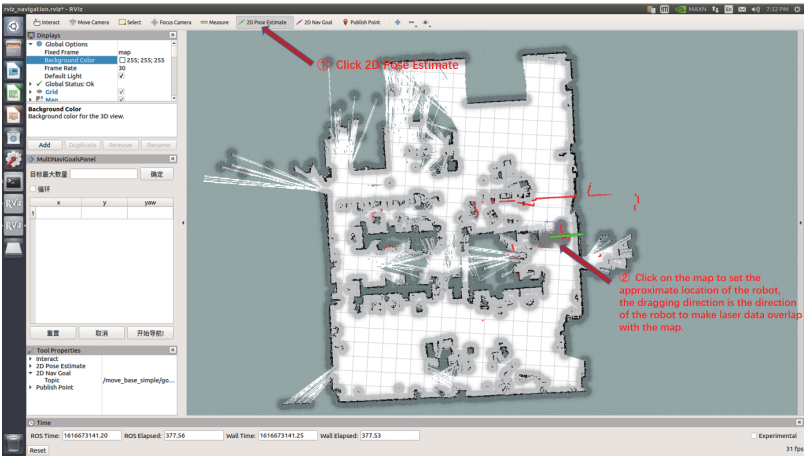
(1) Start the lidar and publish the coordinate transformation of base_link-><laser_link>

```
roslaunch scout_bringup open_rslidar.launch
```

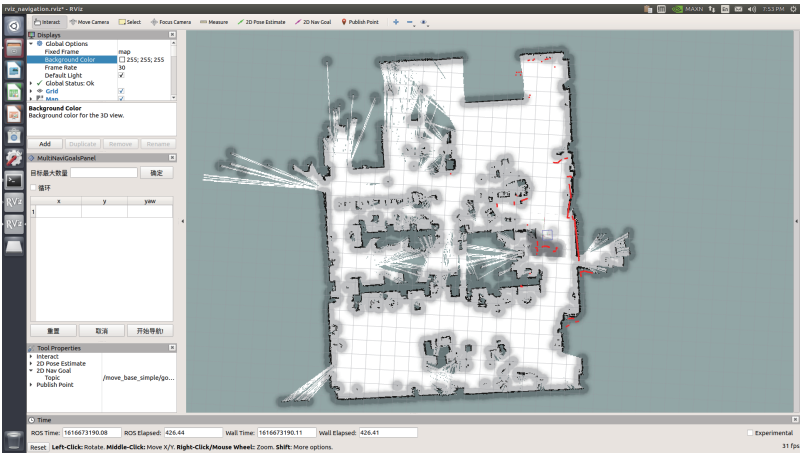
(2) Start move_base navigation

```
roslaunch scout_bringup navigation.launch
```

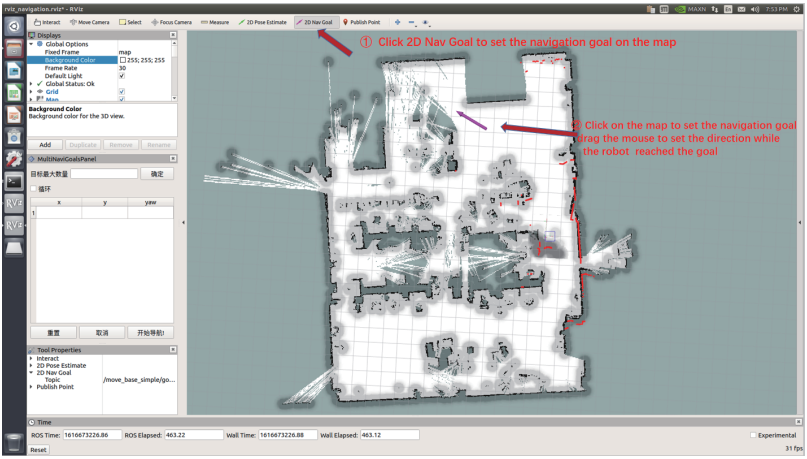
(4) Regulate the actual position of the robot on the map displayed in rviz, and regulate the chassis rotation by publishing an approximate position with the handle. When the laser shape overlaps with the scene shape in the map, the regulation is complete.



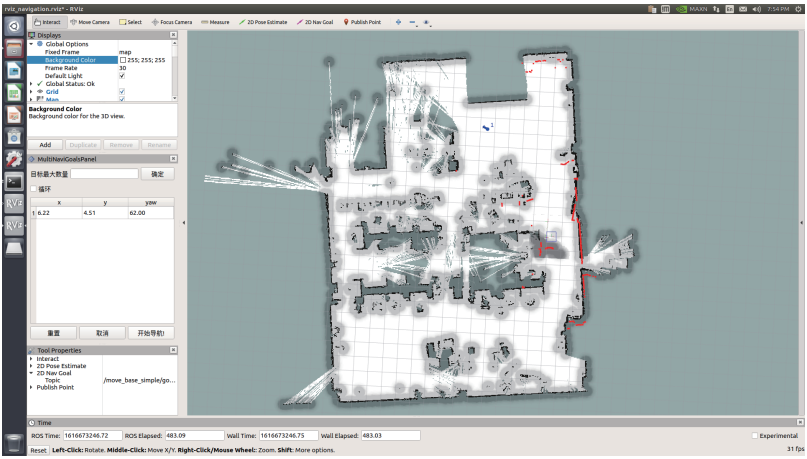
After setting the positioning, the laser shape and the scene shape in the map have basically overlapped, the means the regulation is complete. As the picture shown below.



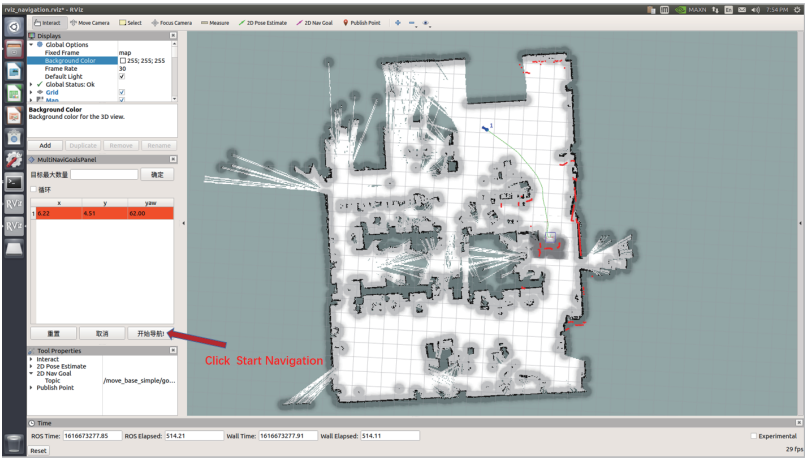
(4) Set multiple target points on the multi-point navigation plug-in in the left column, click Start Navigation, and switch the SWB to command control mode.



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.



Génération ROBOTS

Distributeur officiel

sales@generationrobots.com

+33 5 56 39 37 05

www.generationrobots.com