miiboo

miiboo robot

Build Your Dream

Product manual

V 3.0

【About us】

website	*www.miiboo.cn*
github	*www.github.com/miiboo/*
amazon	*www.amazon.com/dp/B07X2HQ23D*
email	*robot4miiboo@163.com*

Catalog

1. Product Profile

- About miiboo robot
- Application area
- Why ROS
- what learn from miiboo
- Performance parameter
- Function Lists
- Shipping details

2. Product analysis

- Structure layout
- Hardware architecture
- Motor drive board
- Motor with Encoder
- Lithium battery
- Raspberry pi 3B
- Software architecture
- ROS Code Organization Form on Robot
- Function display

3. Product usage

- Connect WiFi to robots
- ROS Network Communication
- SLAM mapping
- Autonomous navigation
- Voice Interaction

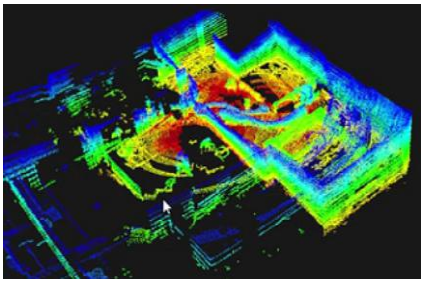
About miiboo robot

Miiboo robot development platform serves the vast number of robot enthusiasts, creators, enterprises, universities, research institutions. Our team is dedicated to the core technology research and development of chassis hardware and software solutions, sensor-driven and data fusion, SLAM mapping, navigation, path planning, robot remote monitoring, speech recognition, speech synthesis and voice interaction.

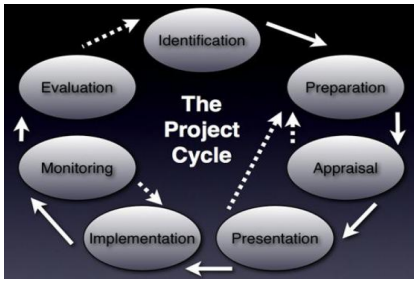
Application area

ROS Initial Learning Platform

Practice of Science and Technology Competition



Research on SLAM Algorithms



Enterprise Pre-development

Secondary development

Why ROS

ROS (Robot Operating System) is a software platform for robots. Provide hardware abstraction, device driver, common function calls, inter-process messaging, and package management. It also provides code tools and library functions to capture, compile, write, and cluster computing.

what learn from miiboo

.Assembly Robot

.Development Board Connection Exercise

.Program example experience

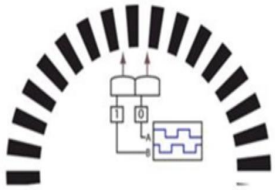
.Programming Advanced Learning

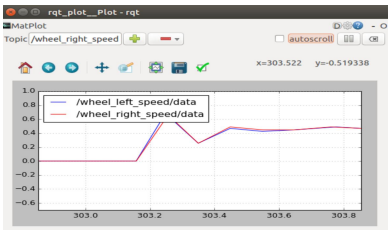
.Improved algorithm

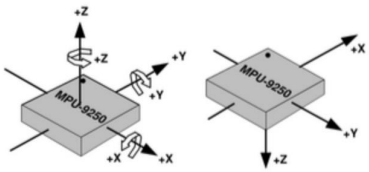
Performance parameter

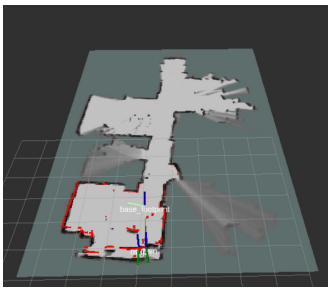
Mechanical parameters	size	300mm*220mm*230mm
	Driving mode	2WD drive
	power wheel	65mm
	Cardan wheel	45mm
	Net weight	4kg
	Maximum load	10kg
	Maximum speed	0.5m/s
Battery parameters	Capacity	12V 4000mAh
	Charging	2 hours full using 12V/2A Charger
	Endurance	5h
Hardware parameters	Raspberry pi 3B	ARM-Cortex-A53,1GB RAM,32GB flash SD-card
	Motor	12V DC deceleration motor,AB encoder 616 line
	Motor drive board	STM32 MCU、TB6612 Driver Chip
	Lidar	5kHZ sampling,8HZ scan、10m maximum range
	IMU	9 axis,Self-calibration,Built-in data fusion
	Camera&Microphone	640*480 pixel,4 Microphone array
	Sound	Dual Channel,USB sound card
OS	Raspberry pi 3B OS	Ubuntu-mate-16.04,ROS-kinetic
	Virtual Machine OS	Ubuntu-16.04,ROS-kinetic
perform	positioning accuracy	5cm
	odometry accuracy	1m(error < 1%)
	Voice Interaction	ASR,TTS,Continuous Interaction
Interface	hardware interface	USB2.0,UART,wifi
	software interface	ROS API,rosjava API

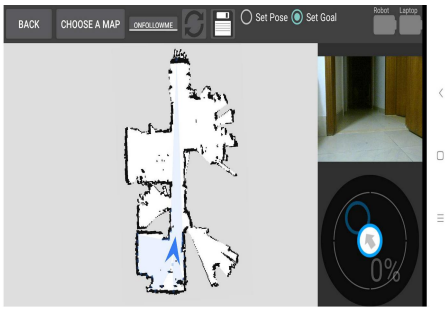
Function Lists









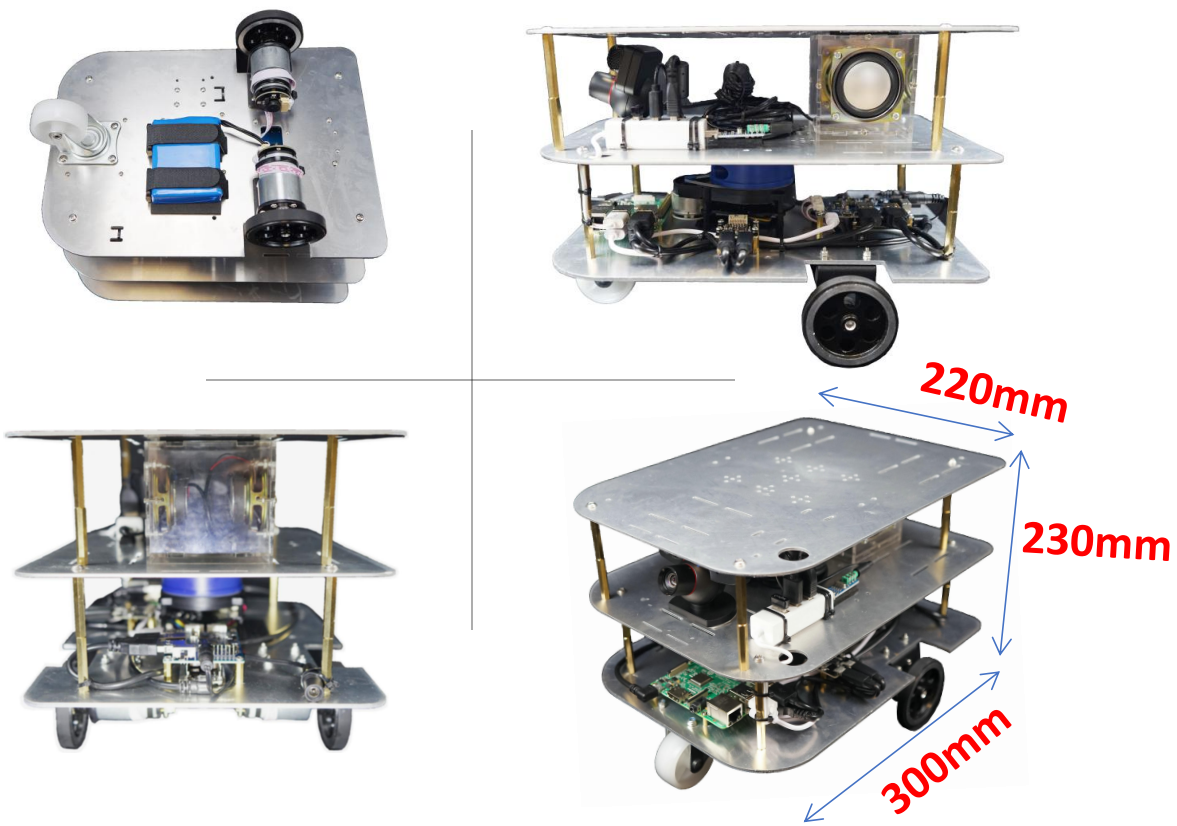


01	ROS odometry data feedback
02	Odometry calibration
03	Line speed and angular velocity concurrency control
04	Dynamic adjustment of motor PID parameters
05	9-axis IMU self-calibration and attitude fusion
06	Google-cartographer SLAM based on odometry,lidar,IMU data fusion
07	Save and convert the cartographer map with one click
08	Autonomous navigation based on ros-navigation
09	Teb efficient path planning and dynamic obstacle avoidance
10	Global localization with AMCL
11	Multi-target point navigation and patrol
12	Remote video surveillance with camera
13	Start-up ROS node
14	Android APP
15	ASR,TTS
16	Continuous Interaction
17	ROS API,rosjava API
...	New features are continuously updated...

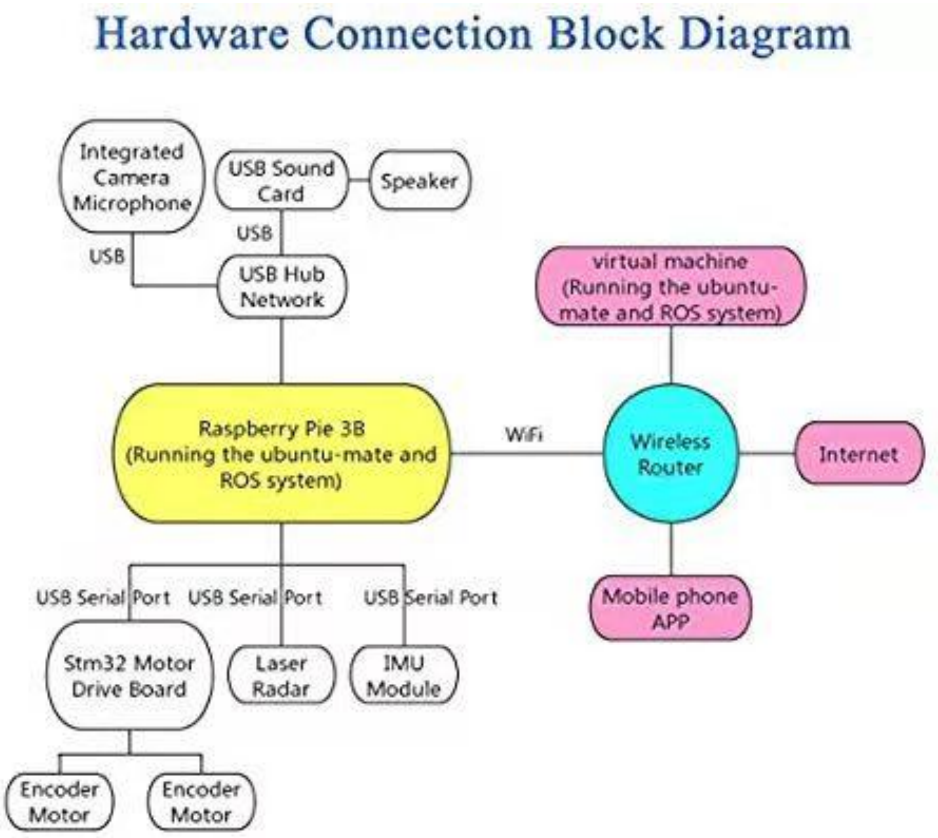
Shipping details

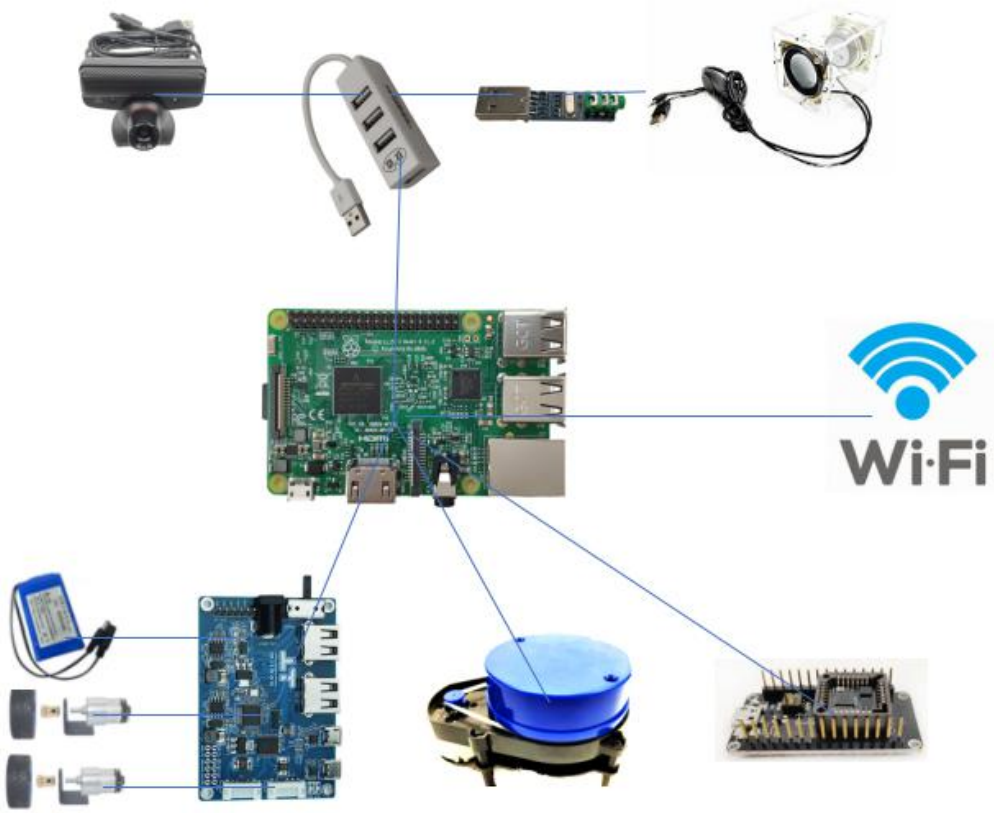
name		specification	number
Robot machine	Aluminum alloy plate	Bottom/Middle/Top	1
	Cardan wheel	45mm	1
	Motor kit	Motor,encoder,wheel	2
	Lithium battery	12V 4000mAh	1
	Charger	12.6V 2A	1
	Motor drive board	stm32 MCU	1
	Lidar	EAI-X4	1
	IMU	9axis,UART output	1
	Raspberry pi 3B	modle 3B	1
	SD card with OS	32GB,Built-in System Mirror	1
	USB-HUB	4 ports,USB2.0	1
	USB sound card	USB2.0	1
	sound	Dual Channel	1
	Camera&Microphone	640*480 pixel,4 Microphone array	1
	Other	Wire rod ,Installation accessories	Some
All codes for robot		Built-in SD card of Raspberry pi 3B	

Structure layout

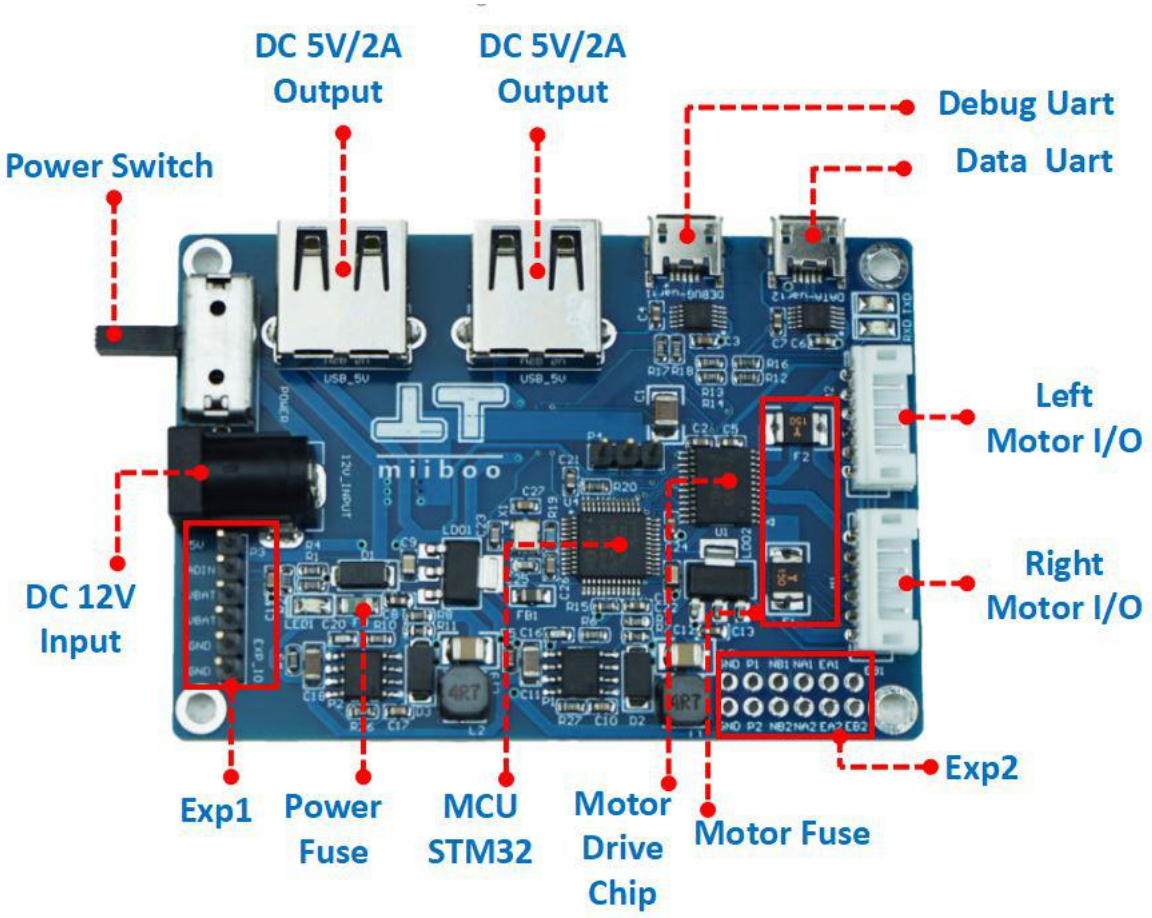


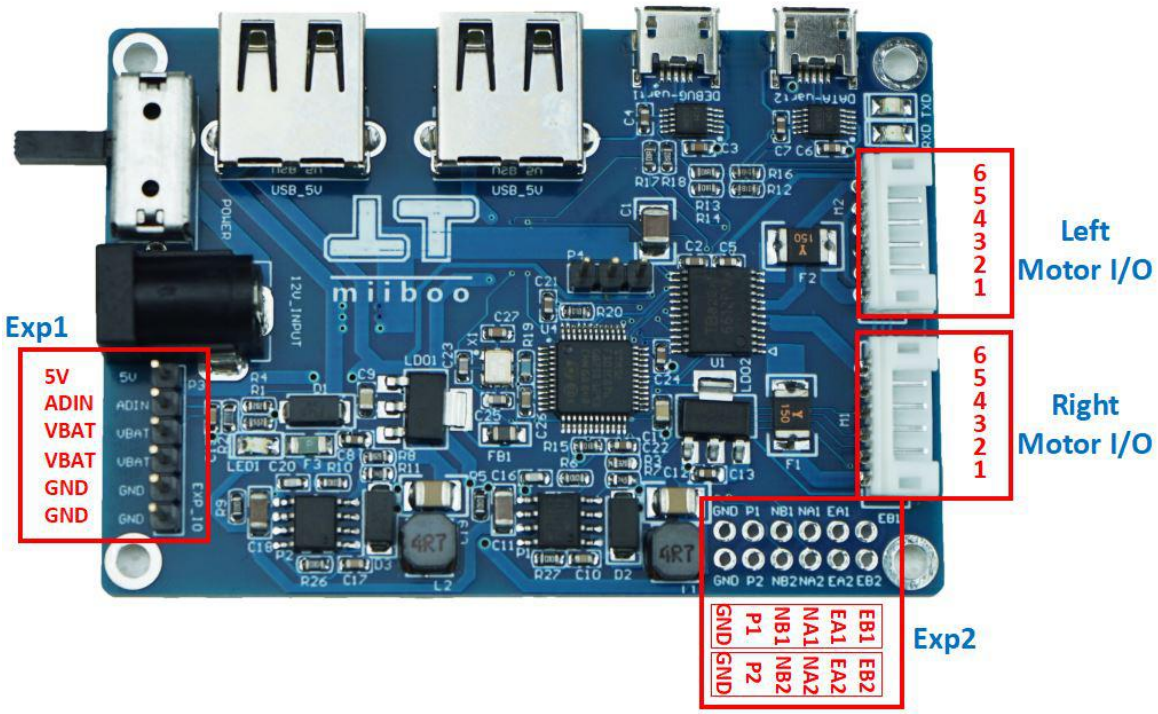
Hardware architecture





Motor drive board



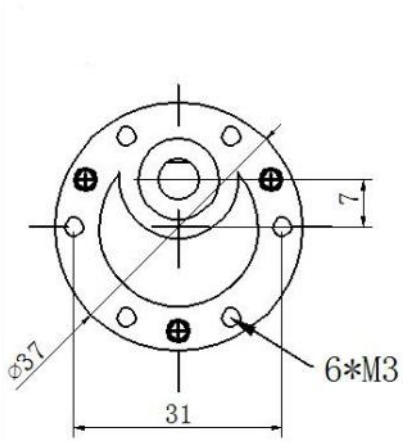


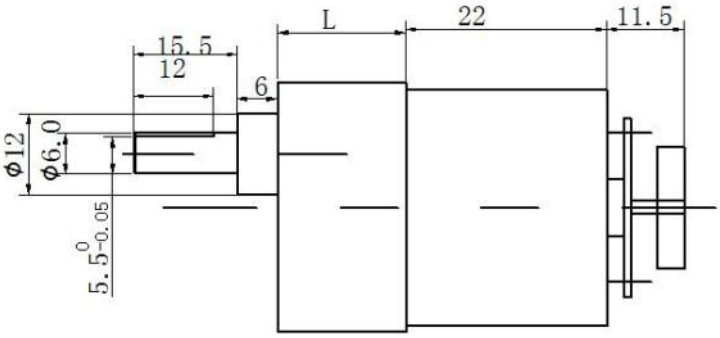
2.Product analysis

name	type	value	Description
DC 12V Input	DC5.5*2.1 circle	12V(9~12.6V)	power input
DC 5V/2A Output	USB type-A	5V/2A	5V output
DC 5V/2A Output	USB type-A	5V/2A	5V output
Debug Uart	microUSB	CH340 UART	Debug UART
Data Uart	microUSB	CH340 UART	Data UART
Left Motor I/O	PH2.0-6P	1: M+ （12V motor） 2: Encoder GND 3: Encoder A （3.3V/5V） 4: Encoder B （3.3V/5V） 5: Encoder VCC （5V） 6: M- （12V motor）	Left GM37 Motor I/O
Right Motor I/O	PH2.0-6P	1: M+ （12V motor） 2: Encoder GND 3: Encoder A （3.3V/5V） 4: Encoder B （3.3V/5V） 5: Encoder VCC （5V） 6: M- （12V motor）	Right GM37 Motor I/O
Exp1	6Pin I/O 2.56mm	5V: 5V output ADIN: NC VBAT: voltage VBAT: voltage GND: Ground GND: Ground	power expand
Exp2	6*2Pin I/O 2.56mm	GND: Ground P1: PWM （3.3V） NB1: direction （3.3V） NA1: direction （3.3V） EA1: Encoder A （3.3V） EB1: Encoder B （3.3V） GND: Ground P2: PWM （3.3V） NB2: direction （3.3V） NA2: direction （3.3V） EA2: Encoder A （3.3V） EB2: Encoder B （3.3V）	High-power motor expand

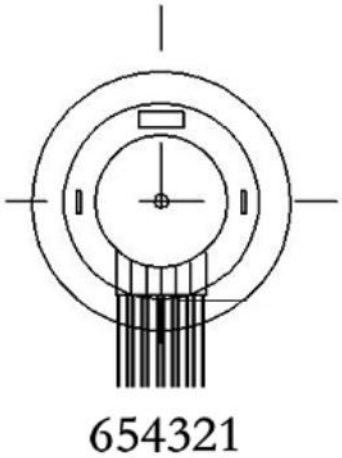
Motor with Encoder

The 12V DC deceleration motor with encoder provides power for the chassis. The motor kit includes deceleration motor, coupling, tire, bracket, etc. Eleven pulses are generated by the motor rotating in one circle. The resolution of the encoder is equal to 11* deceleration ratio. The resolution of the deceleration motor with 56 deceleration ratio is 616 lines.





1: M+ （12V motor）
2: Encoder GND
3: Encoder A （3.3V/5V）
4: Encoder B （3.3V/5V）
5: Encoder VCC （5V）
6: M- （12V motor）

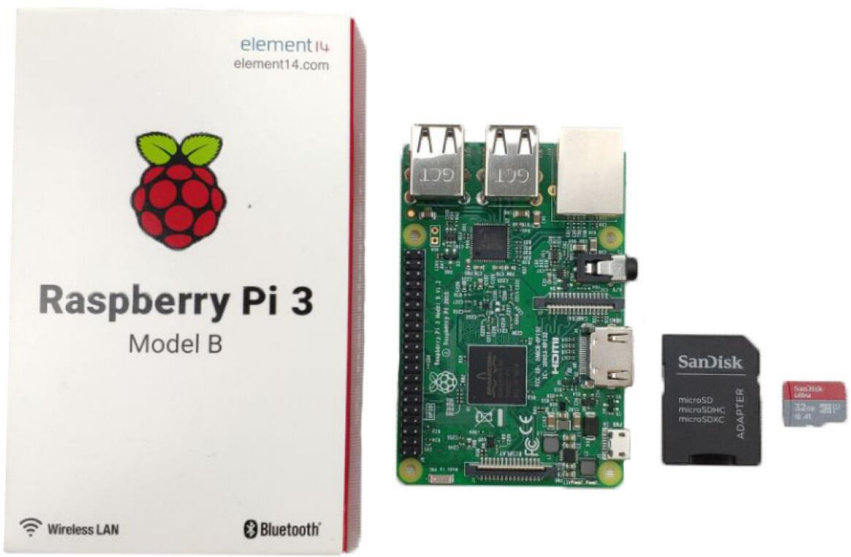


Lithium battery

The 12V 4000mAh rechargeable lithium battery provides power for the chassis and lasts for up to 5 hours. Equipped with 12.6V 2A special charger, it can be filled in 2 hours.

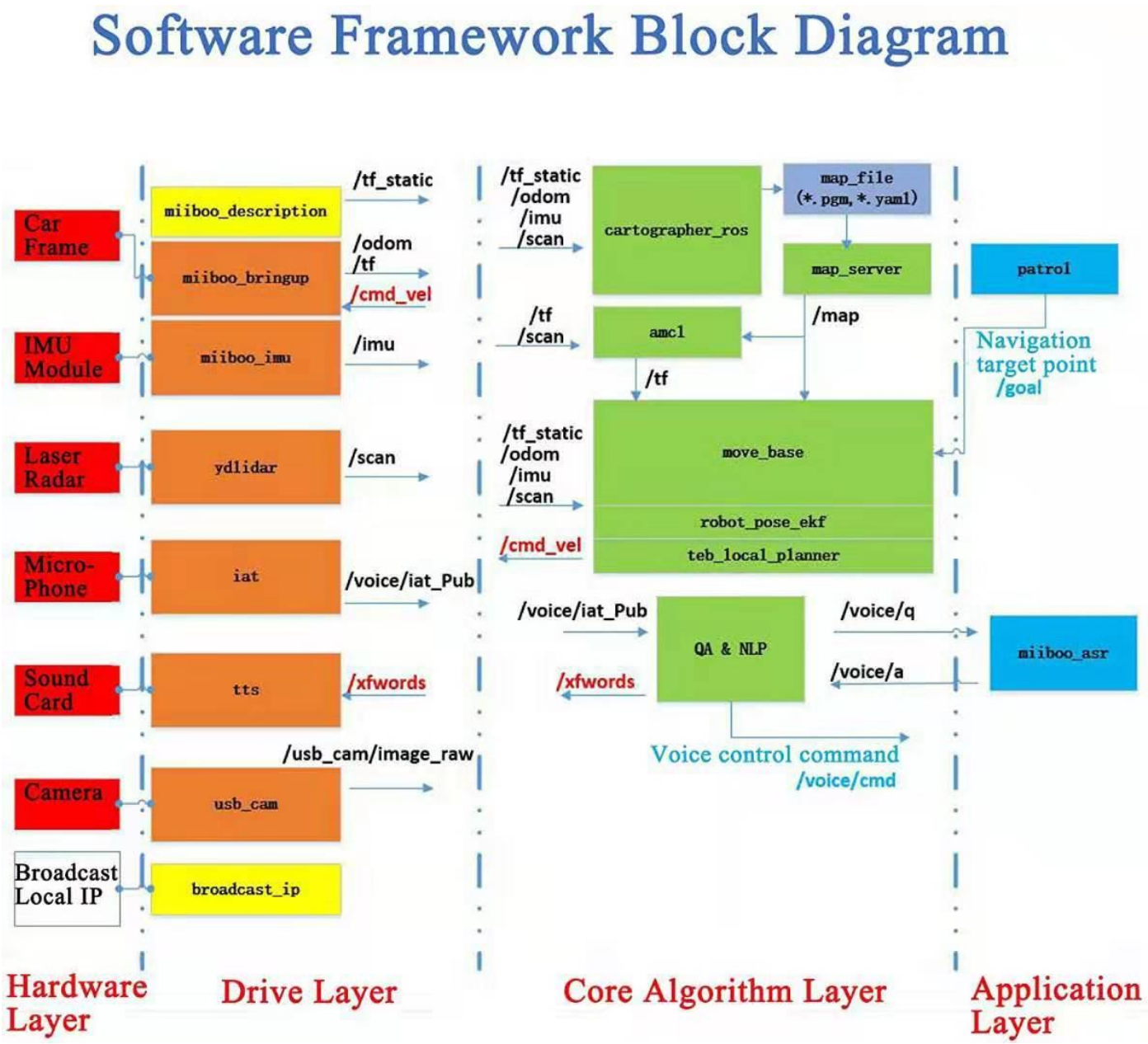
Raspberry pi 3B

The main control of the robot is the raspberry pi 3B with high cost performance. The SD card has recorded the complete source code of ubuntu-mate system, ROS-kinetics, robot configuration, sensor driver and SLAM navigation. Insert SD card, access raspberry pie 3B power supply, you can experience immediately.



Software architecture

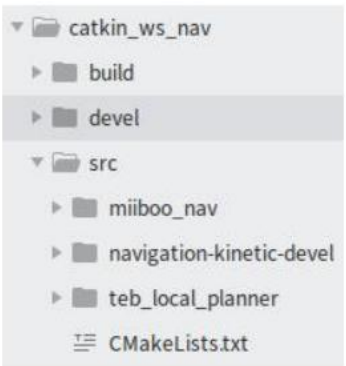
The software architecture of the robot is divided into four layers: hardware layer, driver layer, core algorithm layer and application layer.



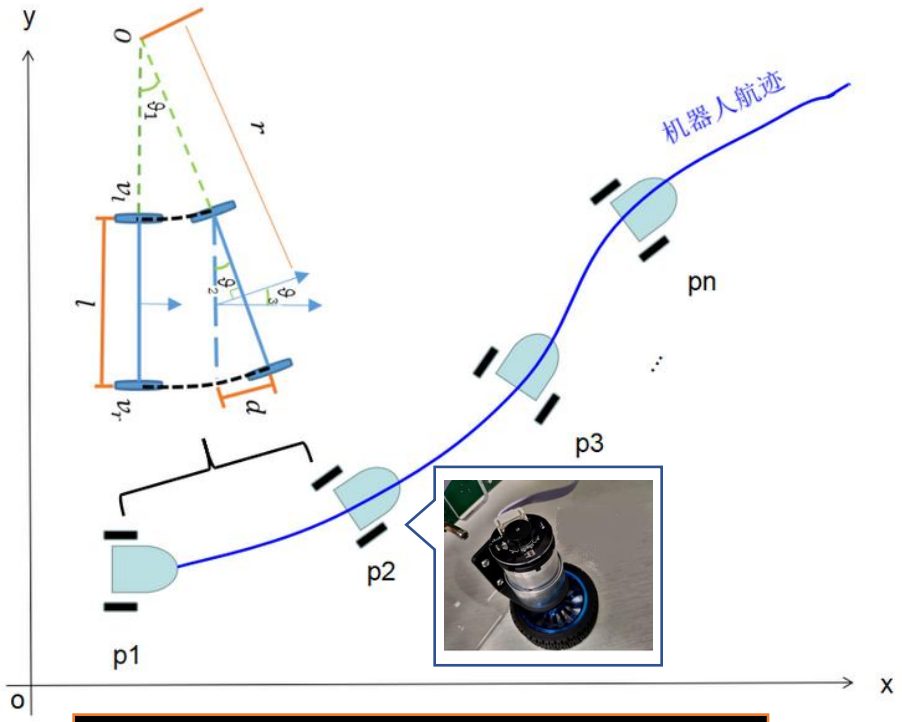
ROS Code Organization Form on Robot

All ROS codes on the robot are stored in the raspberry pi 3B supporting SD card system, insert SD card, access the raspberry pie 3B power supply. After entering the raspberry pie system and switching to the /home/ubuntu directory, you can find four ROS workspaces: catkin_ws, catkin_ws_carto, catkin_ws_nav and catkin_ws_apps.

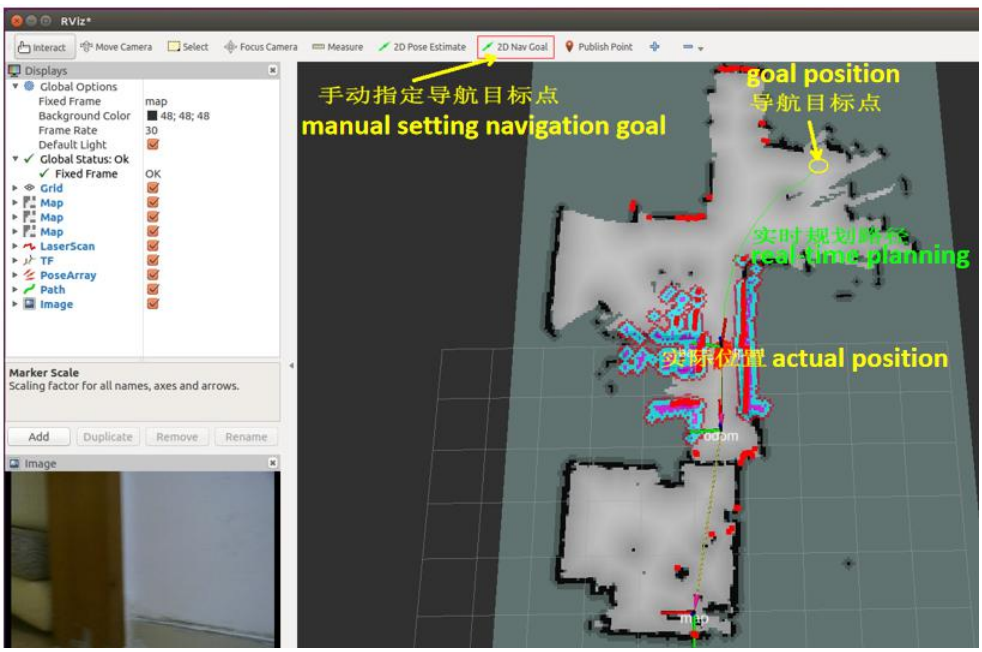
The ROS driver packages of each sensor of the robot are stored in catkin_ws, the SLAM packages based on cartographer are stored in catkin_ws_carto, the navigation packages are stored in catkin_ws_nav, and the high-level application packages are stored in catkin_ws_apps.



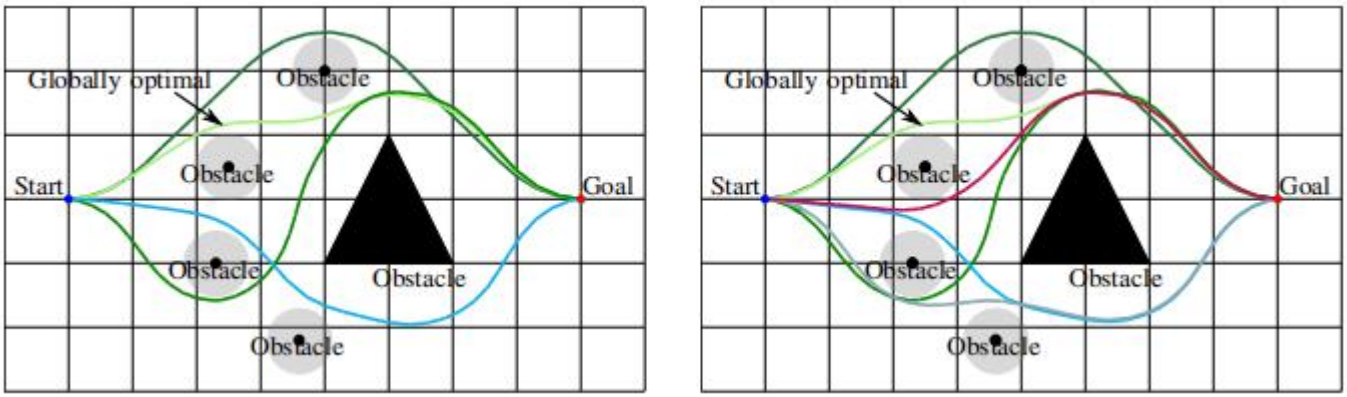
Function display



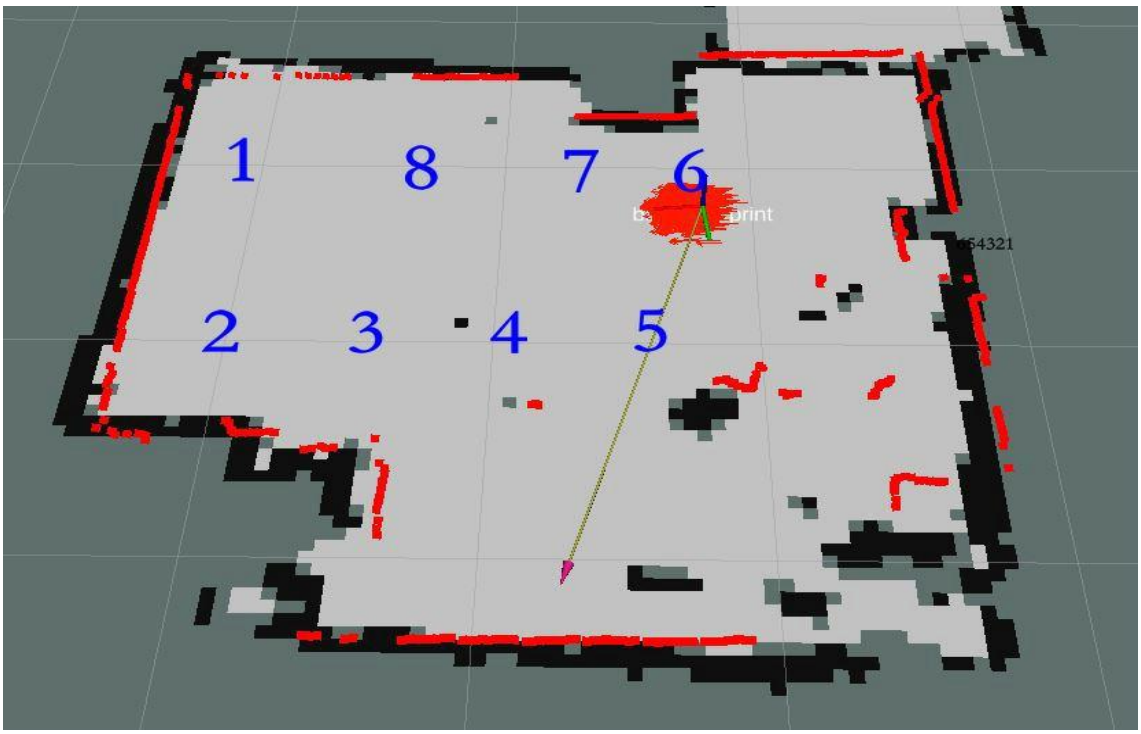
ROS odometry data feedback



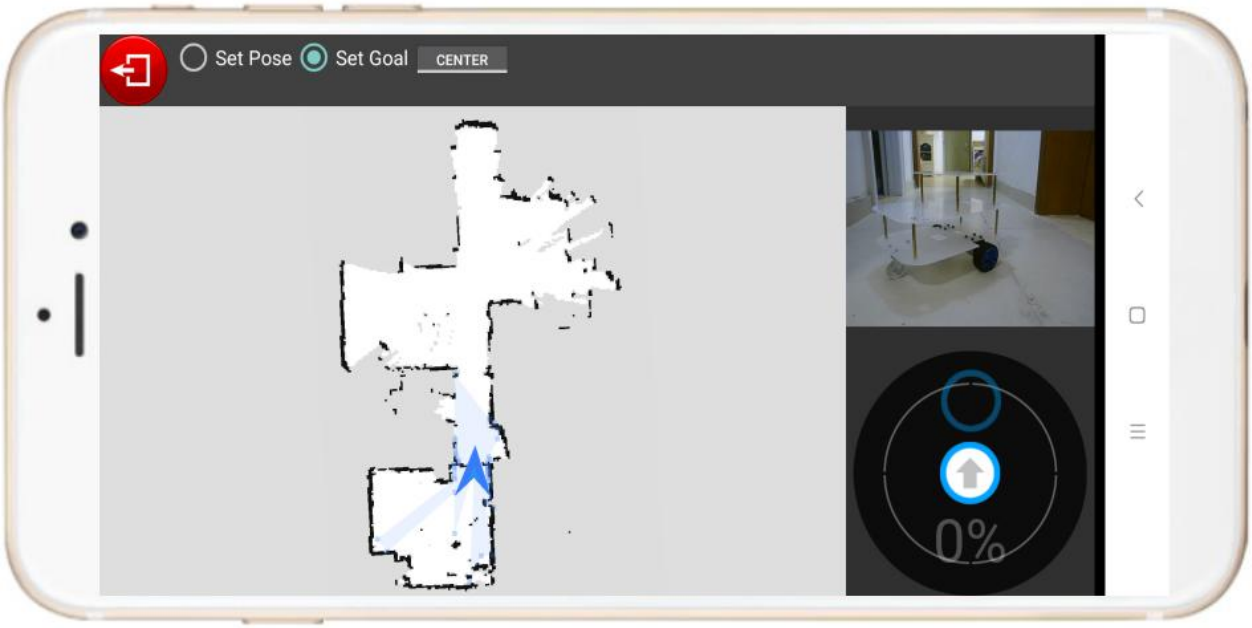
SLAM & Navigation



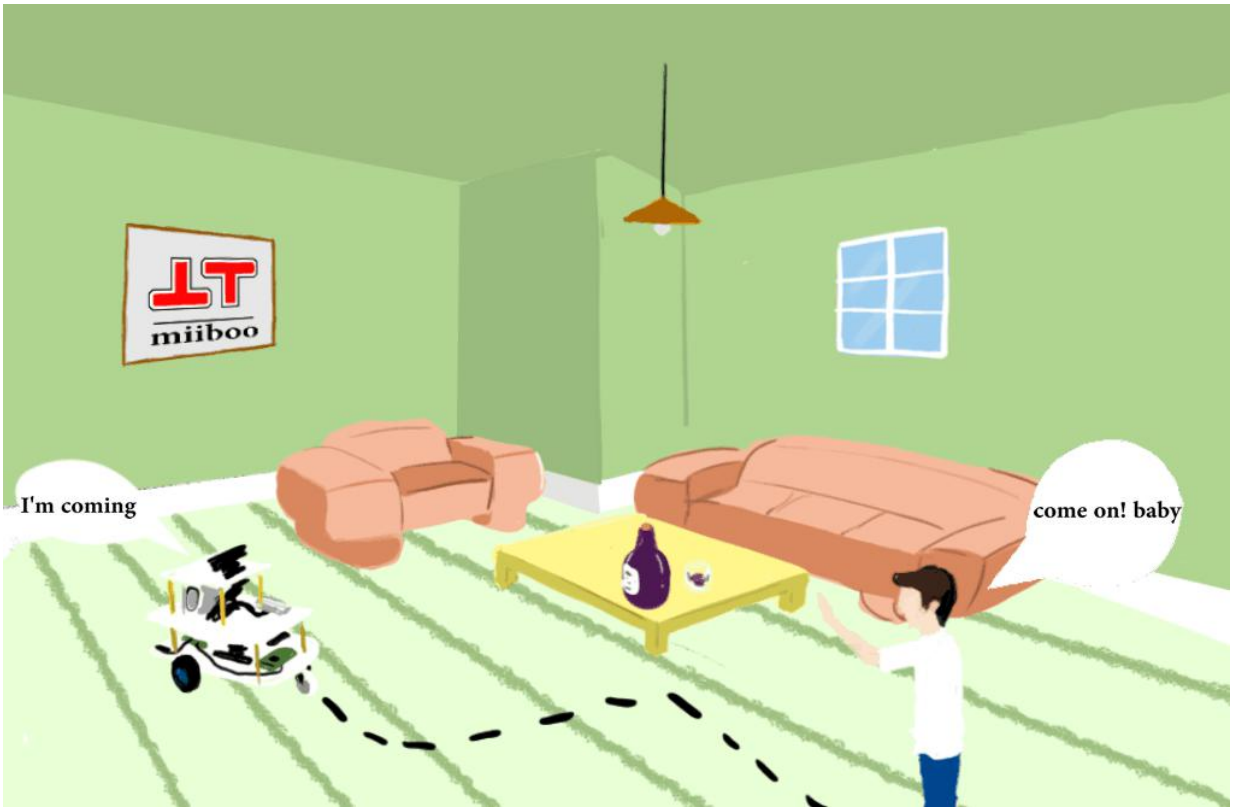
Dynamic obstacle avoidance



Multi-target point patrol



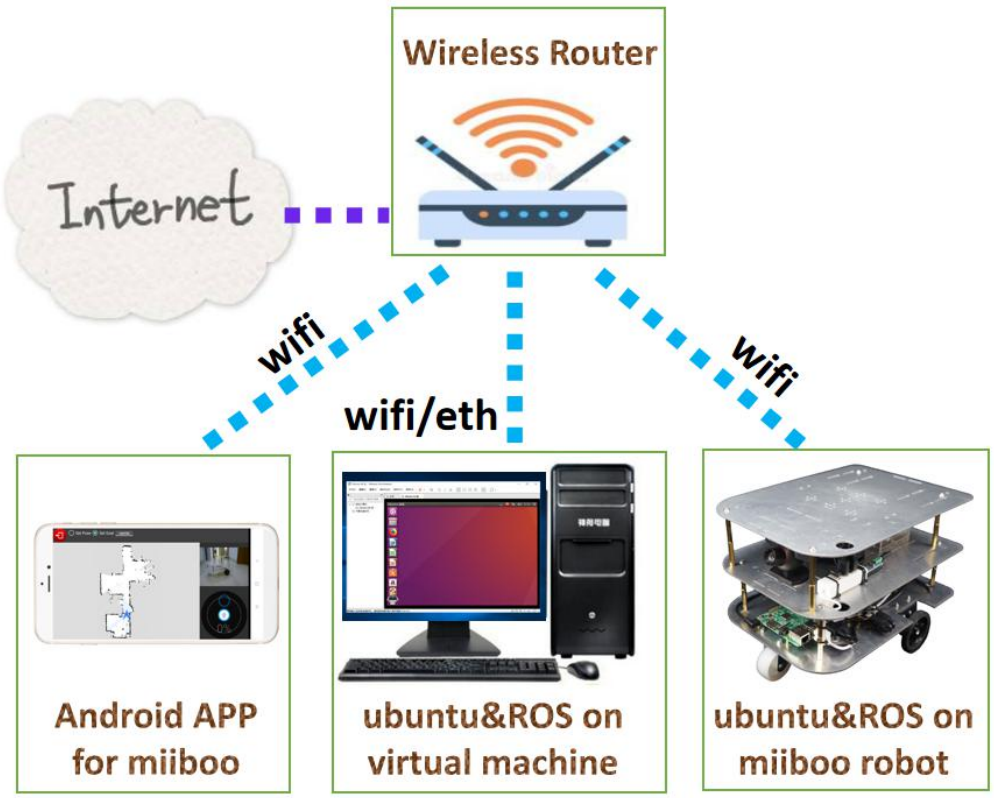
Android APP



Voice Interaction

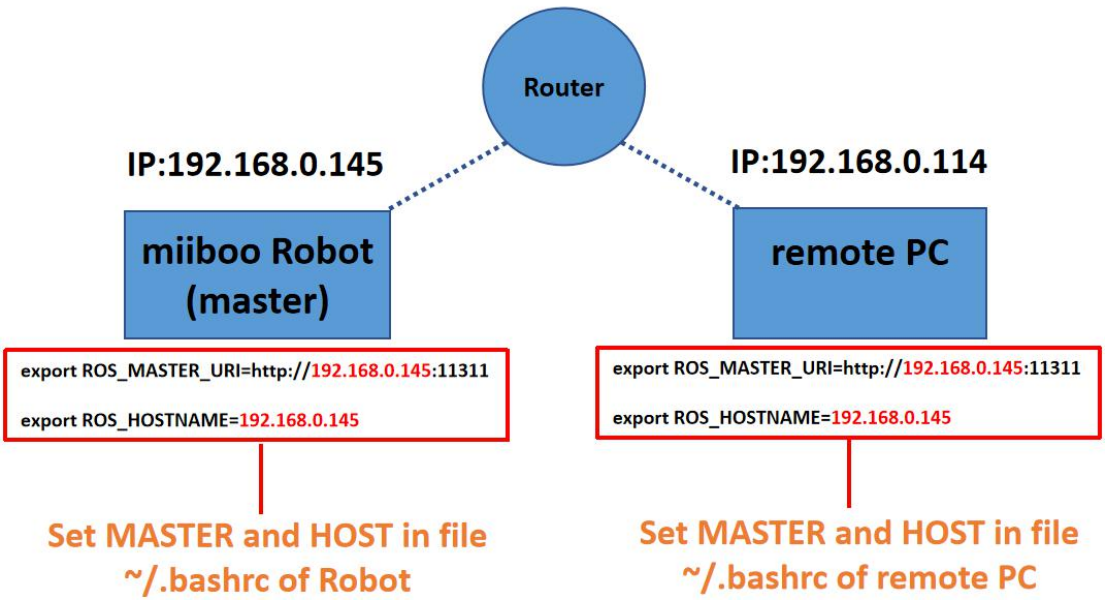
Connect WiFi to robots

Take out the miiboo robot from the box and connect the prepared wireless keymouse and HDMI display to the robot.Then you can turn on the power switch to turn the robot on.Enter the system desktop of Ubuntu mate,Click on the WiFi style icon in the top right corner of the desktop to make a WiFi connection.After WiFi connection is completed, IP address needs to be set to static mode.



ROS Network Communication

The management of ROS network communication adopts star connection mode, that is, each host must specify the same host as MASTER, and declare its own host as HOST. Therefore, two environment variables, MASTER and HOST, need to be set on each host participating in ROS network communication. The values of MASTER and HOST are the real IP addresses of hosts in LAN. In the example, the robot end is designated as MASTER, and the robot end HOST and the remote workbench end HOST participating in ROS network communication are declared. Although the mobile terminal also participates in the ROS network communication, there is an automatic mechanism for setting MASTER and HOST on the mobile APP, so there is no need to set it manually.



SLAM mapping

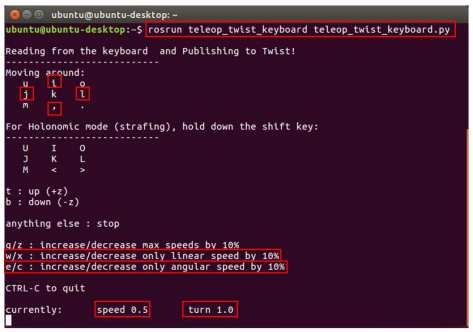
Start all sensors on the robot:

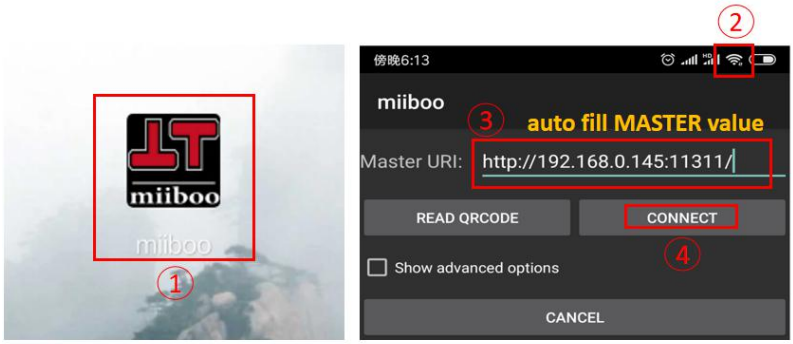
```
$ roslaunch  miiboo_bringup  miiboo_all_sensor.launch
```

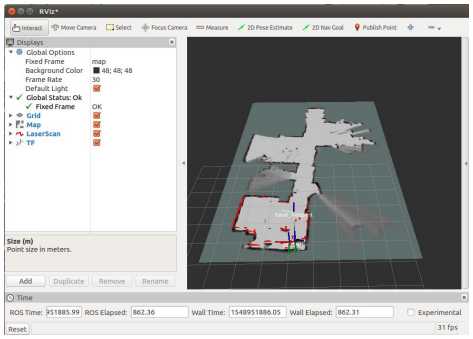
Start cartographer mapping:

```
$ roslaunch  cartographer_ros  miiboo_mapbuild.launch
```

Mapping with Remote Controller:







Save map:

```
$ rosservice  call  /write_state  /home/ubuntu/map/carto_map.pbstream
```

Convert format of map:

```
$ roslaunch cartographer_ros miiboo_pbstream2rosmap.launch \
  pbstream_filename:=/home/ubuntu/map/carto_map.pbstream \
  map_filestem:=/home/ubuntu/map/carto_map
```

Autonomous navigation

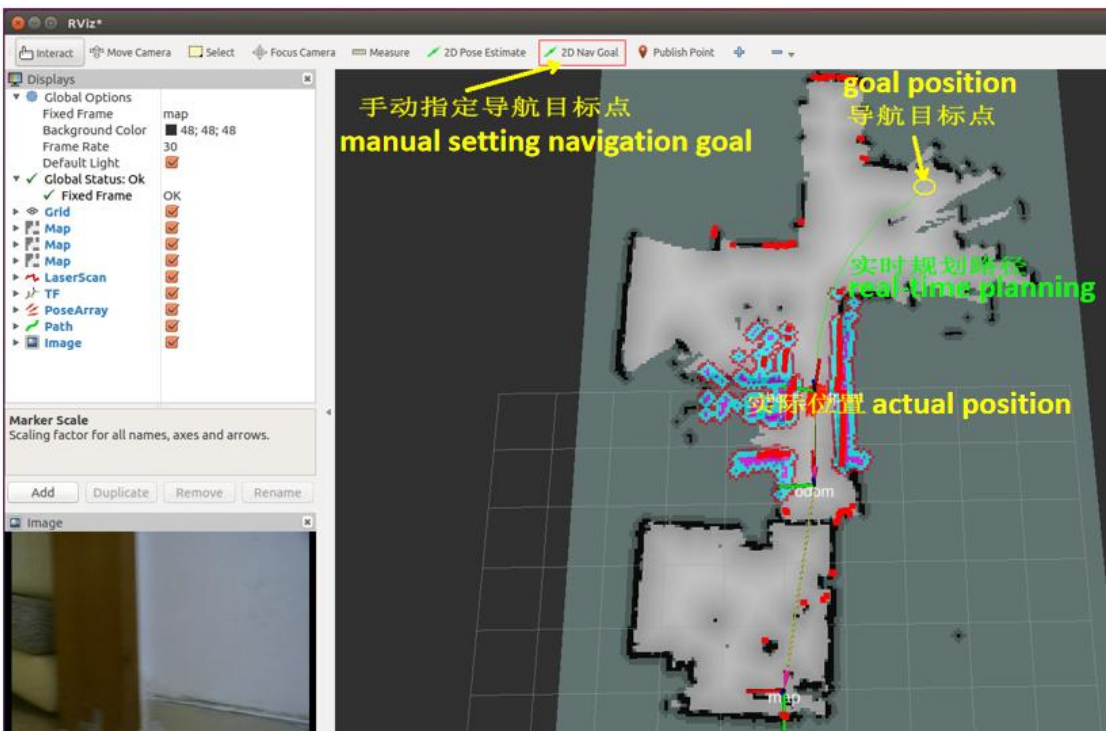
Start all sensors on the robot:

$ roslaunch miiboo_bringup miiboo_all_sensor.launch

Start autonomous navigation:

$ roslaunch miiboo_nav miiboo_nav.launch

Send Navigation Target Point:



Voice Interaction

```
$ roslaunch  miiboo_asr  xf.launch
```