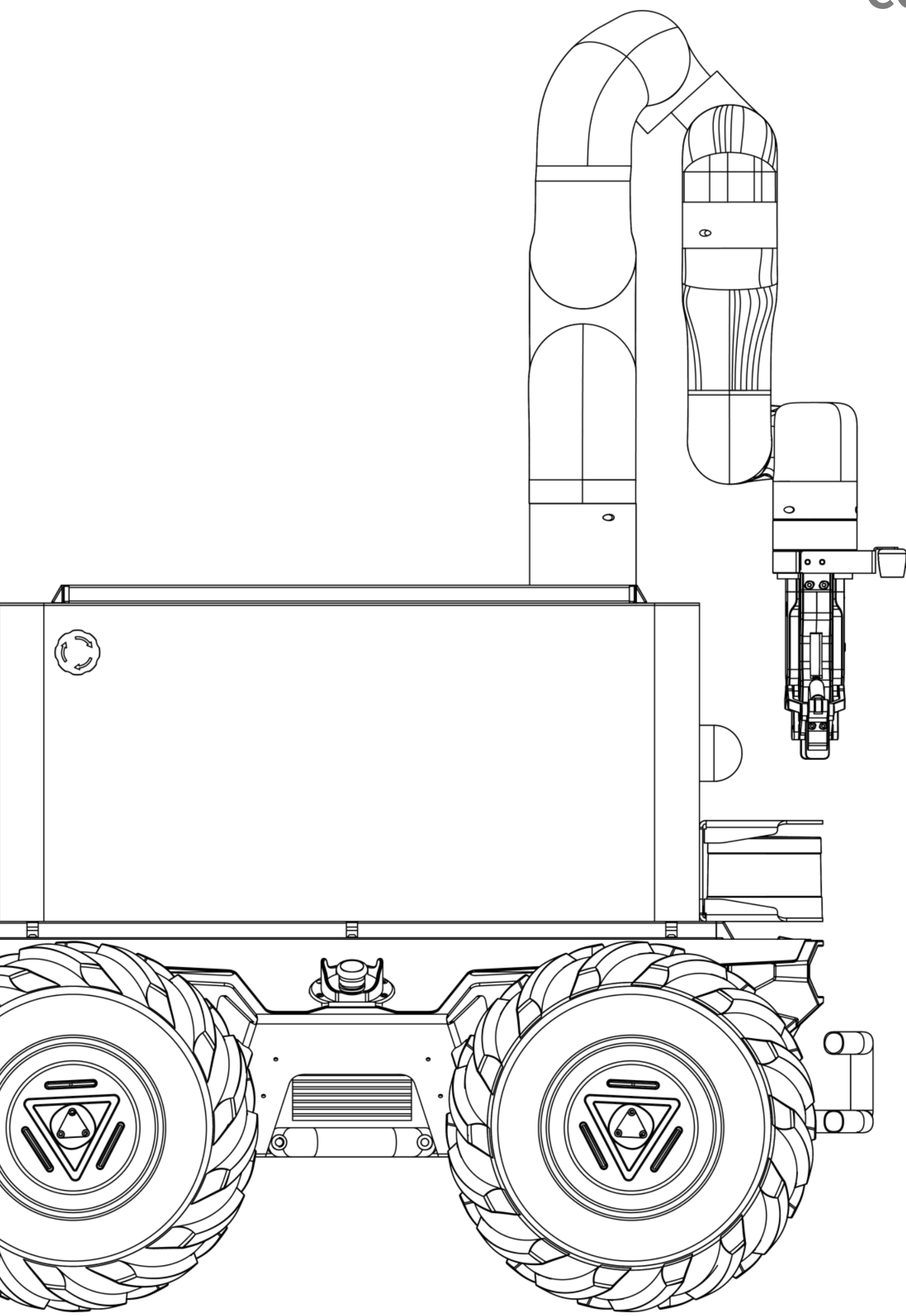
AGILE·X Robotics

COBOT KIT
用户指导手册

COBOT KIT User Manual



V1.0 2023.04

2 产品简介

Agile复合移动机器人系列开发套件是松灵机器人专为行业科研教育应用客户研发开发的高级开发套件，面对科研教育对机器人产品越来越灵活的，该套装基于松灵机器人ROS生态体系统，集成高性能工控、高精度LiDAR、多传感器、多自由度机 械臂、视觉感知搭配于一身。套件包含了多线激光雷达自主导航、机械臂moveIt运动控制规划、视觉识别、机械臂自主抓 取等功能于一体，解决客户在面对移动机器人在复杂应用场景下的技术复杂度高、集成难度高的特征，给客户带来极致的用 户体验。产品包含完整的技术手册与配套的技术文档、代码采用全开源的形式，降低客户的应用和学习难度。产品套件可以 广泛用于农业、智能制造、教育实训、科研探索等方向。

1 Product Introduction

The Agile Composite Mobile Robot Development Kit is an advanced development kit designed by Agile Robotics specifically for industry, research, and educational applications. In response to the increasing demand for flexible robotics products in these areas, this kit is built on the Agile Robotics ROS ecosystem and integrates high-performance industrial control, high-precision LiDAR, multiple sensors, multi-degree-of-freedom manipulator arm, and visual perception in one package. The kit includes autonomous navigation with multi-line LiDAR, motion control planning with manipulator arm moveIt, visual recognition, and autonomous grasping, providing customers with an ultimate user experience to solve the high technical complexity and integration difficulty of mobile robots in complex application scenarios. The product includes a complete technical manual and supporting technical documents, and the code is open source, making it easier for customers to apply and learn. The product kit can be widely used in agriculture, intelligent manufacturing, education and training, and scientific research exploration, among other fields.

3 主要配置清单以及参数说明

3.1 主要配置

2 Main Configuration List and Parameter Description

2.1 Main Configuration

套件版本 Kit Version	Cobot Kit
底盘移动平台 Chassis Mobile Platform:	SCOUT 2.0
机械臂 Gripper	Xarm 5
机械臂夹爪 Robotic Arm Gripper	大寰AG 95 DH AG 95
工控机 Industrial Personal Computer:	x-7010(i7 16G 256SSD)
视觉传感器 Visual Sensor:	RealSense D435
激光雷达传感器 LiDAR Sensor:	RS-Helios-16P
屏幕 Screen:	高清显示屏 High-Definition Display Screen
路由器 Router:	B316 4G路由器 Router

3.2 主要配件介绍

- SCOUT 产品介绍

SCOUT2.0是一款全能型行业应用UGV（UNMANNED GROUND VEHICLE）。它是一款采用模块化、智能化设计理 念的多功能模块化的行业应用移动机器人开发平台， 具有强大载荷能力和强劲动力系统的它具有广泛的应用 领域。立体相机、激光雷达、 GPS、 IMU、 机械手等设备可选择加装至SCOUT2.0作为扩展应用。SCOUT2.0 可被 应用到无人巡检、安防、科研、勘探、物流等领域。

2.2 Kit Accessories Introduction

SCOUT Product Introduction

SCOUT 2.0 is a versatile Unmanned Ground Vehicle (UGV) designed for industry applications. It is a modular and intelligent design platform with a strong load capacity and powerful drive system, making it suitable for a wide range of applications. The platform can be extended with devices such as 3D cameras, LiDAR, GPS, IMU, and robotic arms. SCOUT 2.0 can be used for unmanned inspections, security, research, exploration, logistics, and other industries.

参数类型 Type	项目 Items	指标 Parameters
机械参数 Mechanical	长×宽×高 Dimensions （mm）	930 X 699 X 348
	轮距 Axle Track（mm）	498
	前 / 后轮距 Front/rear track (mm)	582/ 582
	车体重量 Body weight	65-68
	电池类型 Battery type	锂电池24V 30Ah Lithium battery 24V 30Ah
	电机 Motor	直流无刷 4 X 400W Brushless DC 4 X 400W
	减速箱 Gearbox	1:40
	驱动形式 Driving type	四轮独立驱动 Four wheel independent drive
	悬架 Suspension	单摇臂独立悬架 Single swing arm independent suspension
	转向 Steering type	四轮差速转向 Four wheel differential steering
	安全装备 Safety equipment	伺服刹车/防撞管 Servo brake/anti-collision pipe
性能参数 Performation	空载最高车速 No-load maximum speed (m/s)	1.5
	最小转弯半径 Minimum turning radius (mm)	可原地转弯 Can turn in place
	最大爬坡能力 Maximum gradeability (°)	30°
	最小离地间隙 Minimum ground clearance (mm)	135
控制参数 Control	控制模式 Control mode	遥控控制 Remote controller mode 控制指令模式 Command control mode
	遥控器 remote controller	2.4G/极限距离1KM 2.4G/limit distance 1KM
	通讯接口 Communication	CAN / RS232

- Xarm机械臂介绍

Xarm系列机器人是松灵机器人根据科研教育行业严选出来的一款工业级轻量级协作机器人产品，产品采用关节模块化设计，使用面向开发者层面的机器人系统，用户可通过此界面实时观察机器人的运行状态，对机器人进行诸多控制设置，也可脱机进行离线仿真，极大地提升了实际应用的工作效率.。同时我们根据科研教育行业的特点，适配ROS，支持Moveit等开源方案支持。

Introduction of Xarm robotic arm

The Xarm series of robots is a lightweight industrial collaborative robot product selected by Agile Robotics specifically for the scientific research and education industry. The product adopts joint modular design and uses a robot system oriented towards developers. Users can observe the real-time operation status of the robot through this interface, perform various control settings on the robot, and also perform offline simulation, greatly improving the work efficiency of practical applications. At the same time, we have adapted to the characteristics of the scientific research and education industry, and support ROS and open-source solutions such as Moveit.

性能 Performance				
*1.环境温度 Ambient temperature	0-50℃			
功耗 Power consumption	最小 8.4 W，典型 120 W，最大 240 W Minimum 8.4 W, typical 120 W, maximum 240 W			
输入电源 Input power	24 V DC,15A			
物理 Physics				
占地面积 Floor area	Ø 126 mm			
材料 Material	铝合金、碳钎维 Aluminum alloy, carbon fiber			
底座连接器类型 Dock Connector Type	M5*6			
特性 Characteristic				
ISO洁净度等级 ISO cleanliness class	5			
机器人安装 Robot installation	任意角度 Any angle			
I / O接口 I/O interface				
AC / DC控制器 AC/DC controller	8*CI	8*CO	2*AI	2*AO
机械臂末端 End of robotic arm	2*CI	2*CO	2*AI	1*RS485
通讯（机械臂本体）Communication (mechanical arm body)				
通信协议 Communication protocol	自定义			
通讯（AC / DC控制器）Communication (AC/DC Controller)				
通信协议 Communication protocol	TCP-IP / Modbus RTU			
通信方式 Way of communication	Ethernet / RS-485			
机械臂末端通讯 Communication at the end of the robot arm				
通信协议 Communication protocol	RS-485 / Modbus RTU			

*1.机器人的工作温度为0-50℃。当关节在高速连续工作时，请尝试降低环境温度。

- X-7010工控机介绍

X-7010是一款模块化组合的高性能超小工控机，针对移动机器人行业算力要求高，环境要求高的特征而定制的一款工业工控控制电脑。它采用Intel 平台，支持8/9th 35W高速处理器。它采用高效热管的大面积铝鳍和PWM风扇的主/被动双重散热设计，通过模具打造的全铝合金强固机身，保证其长寿命稳定运行。适用于智能机器人、无人驾驶、机器视觉、智慧城市 等高运算要求领域。预装Ubuntu 18.04， ROS Melodic（Full Desktop) 版本，以及一些机器人开发常见得开发环境，实现开机即用。

- 具有巴掌大小紧凑超小机身；
- 支持Intel 8th桌面级高性能CPU；
- 搭载热管与智能风扇的主被动高效散热；
- 支持miniPCIE、NVME等多种加速卡扩展方案；
- 多路超高速专用串口，适配多种雷达应用；
- 多通道USB3.1 Gen2与双千兆网络的高速通讯；
- 坚固的模具成型的铝合金机身，符合车载振动冲击；
- -20~60℃宽温工作；

Introduction of X-7010 Industrial Personal Computer

X-7010 is a high-performance, modular, and ultra-small industrial Personal computer designed for the mobile robot industry that requires high computing power and high environmental requirements. It adopts Intel platform and supports 8/9th 35W high-speed processors. With the efficient heat pipe, large area aluminum fins, and PWM fan's main/passive dual cooling design, and the all-aluminum alloy solid body molded by molds, it ensures long-term stable operation. It is suitable for high-computational-demand fields such as intelligent robots, unmanned driving, machine vision, and smart cities. It comes pre-installed with Ubuntu 18.04, ROS Melodic (Full Desktop) version, and some common robot development environments, achieving plug-and-play functionality.

- It features a compact and ultra-small palm-sized body;
- Supports Intel 8th desktop-level high-performance CPU;
- Equipped with a main and passive high-efficiency heat dissipation system using heat pipes and intelligent fans;
- Supports multiple expansion schemes such as miniPCIe, NVME, etc.;
- Multi-channel high-speed dedicated serial ports compatible with various LiDAR applications;
- High-speed communication through multiple USB 3.1 Gen2 channels and dual Gigabit networks;
- A sturdy all-aluminum alloy body molded by molds that meets the vibration and shock requirements of

automotive applications.

- -20~60℃ wide temperature operation

- 视觉传感器RealSense D435

双目视觉传感器，在机器人视觉测量、视觉导航等机器人行业方向中均有大范围的应用场景和需求，目前我们甄选了了在科研教育行业常见的视觉传感器。英特尔实感深度摄像头 D435 配备全局图像快门和宽视野，能够有效地捕获和串流移动物体的深度数据，从而为移动原型提供高度准确的深度感知。

RealSense D435 Visual Sensor

RealSense D435 is a stereo visual sensor that has a wide range of applications and demands in the robotics industry, such as robot vision measurement and navigation. We have selected this visual sensor which is commonly used in scientific research and education. Equipped with a global shutter and a wide field of view, the Intel RealSense D435 can effectively capture and stream depth data of moving objects, providing highly accurate depth perception for mobile prototypes.

	型号	Intel Realsense D435
基本特征	应用场景	户外/室内
测量距离	约10米	
深度快门类型	全局快门/3um X 3um	
是否支持IMU	支持	
深度相机	深度技术	有源红外
FOV	86° x 57°（±3°）	
最小深度距离	0.105m	
	Model	Intel Realsense D435
Basic Features	Application Scenarios	Indoor/Outdoor
Measurement Range	Approximately 10 meters	
Depth Shutter Type	Global Shutter/3um x 3um	
IMU Support	Yes	
Depth Camera	深度技术Technology	Active Infrared
Field of View FOV	86° x 57°（±3°）	
Minimum Depth Distance	0.105m	

	型号	Intel Realsense D435
深度分辨率	1280 x 720	
最大测量距离	约10米	
深度帧率	90 fps	
RGB	分辨率	1280 x 800
FOV	69.4° × 42.5°（±3°）	
帧率	30fps	
其他信息	尺寸	90mm x 25mm x 25mm
接口类型	USB-C 3.1	

Model: Intel Realsense D435

Depth resolution 1280 x 720

Maximum measurement distance approximately 10 meters

Depth frame rate 90 fps

RGB resolution 1280 x 800

FOV 69.4° × 42.5° (±3°)

RGB frame rate 30 fps

Other information

Dimensions 90mm x 25mm x 25mm

Interface type USB-C 3.1

- 雷达传感器

RS-16 是 RoboSense激光雷达中小巧而先进的一款产品。与同等价位的传感器相比，RS-16 性价比更高，并且保留了 RoboSense 在激光雷达方面比较有突破性的一些主要特点，如实时、360°、三维坐标和距离、附带校准的反射率测量。

RS-16 测量范围可达 100 m，功耗低（约 8 W），重量轻（约 830 g），体积小（Ø103mm x 72mm），具备双重返回性能，这些特点使它成为背包式测量、无人机挂载和其它移动设备的理想选择。

LiDAR Sensor

RS-16 is a compact and advanced product in the RoboSense LiDAR sensor series. Compared to sensors at the same price point, RS-16 has a higher cost performance ratio and retains some of the key features that RoboSense has made breakthroughs in, such as real-time, 360°, three-dimensional coordinates and distance, and reflectivity measurement with calibration.

RS-16 has a measurement range of up to 100m, low power consumption (approximately 8W), light weight (approximately 830g), and small size (Ø103mm x 72mm). It also has dual return performance, making it an ideal choice for backpack measurements, drone mounting, and other mobile devices.

项目	**参数**
最大测距	100m
测距精度	±3cm
扫描速率	单次回波30万点/秒 双回波60万点/秒
垂直视角	-15°~＋15°
垂直角分辨率	2°
扫描频率	5Hz~20Hz
安全等级	Class 1
重量	830g
功耗	8W
电压	9V~18V
工作温度	-10℃~＋60℃

Maximum Range: 100m

Range Accuracy: ±3cm

Scanning Rate: 300,000 points per second for single echo, 600,000 points per second for double echo

Vertical Field of View: -15° + 15°

Vertical Angular Resolution: 2°

Scan Frequency: 5Hz20Hz

Safety Level: Class 1

Weight: 830g

Power Consumption: 8W

Voltage: 9V~18V

Operating Temperature: -10℃ + 60℃

- 机械夹爪

大寰机器人AG 95 电动夹爪拥有两个自适应平行机械 关节手指 （后指代 关节手指每个 关节手指 由多个连杆机构和一个弹簧组成。 关节手指 可以与一个物体进行多达 5 个接触点的接触。 关节 手指采用欠驱动控制方式驱动，使得电机比 关节的总数要少。这种设计简化了抓取的控 制方式，使 关节手指 可以自动适应它们所抓取的物体形状。

Mechanical Gripper

The AG 95 electric gripper from Dahuan Robotics features two adaptive parallel mechanical joint fingers. Each knuckle finger of the posterior knuckle finger consists of multiple linkages and a spring.The joint fingers can make up to 5 contact points with an object. The joint fingers are driven by an underactuated control method, which requires fewer motors than the total number of joints. This design simplifies the control of gripping and allows the joint fingers to automatically adapt to the shape of the object they are gripping.

最大推荐负载	**3-5kg***
手指开合行程（编程可调）	0-95mm
抓持力（编程可调）	45-160N
最快手指开合速度	190mm/s
自身重量	1kg
手指重复定位精度	0.03mm
通讯协议	TCP/IP, USB2.0, RS485, I/O, CAN2.0A, EtherCAT（选配）
工作电压	24V DC±10%
工作温度范围	0~50℃

Maximum recommended load 3-5kg*
Finger opening and closing stroke (programmable) 0-95mm
Gripping force (programmable) 45-160N
Fastest finger opening and closing speed 190mm/s
Self weight 1kg
Finger repeat positioning accuracy 0.03mm
Communication protocol: TCP/IP, USB2.0, RS485, I/O, CAN2.0A, EtherCAT (optional)
Operating voltage: 24V DC±10%
Operating temperature range: 0~50℃

4 实例开发

4.1 开发前准备

4.1.1 下载远程桌面工具

在机器人中的工控机中安装了一个远程桌面的工具，用户需要在自己的笔记本或者电脑上安装相应的工具，通过机器人上面的路由器远程操控机器人上的工控机。

（1）下载安装包

https://www.nomachine.com/download

选择自己笔记本或者电脑对应的操作系统进行下载安装。

（2）使用方法

工控机上的用户：scout，电脑的开机密码：agx。路由器名字：HAIWEI_B316_BD96，路由器的密码和登录密码一致都是12345678。路由器ip：192.168.1.1，工控机ip：192.168.1.100。

首先用自己的电脑或者笔记本找到机器人上面的wifi，输入密码进行连接。

打开下载好的远程连接工具NoMachine，添加一个远程连接：

3 Instance Development

3.1 Preparation before development

3.1.1 Download remote desktop tool

A remote desktop tool has been installed on the Industrial Personal Computer in the robot. Users need to install the corresponding tool on their laptops or computers to remotely control the industrial computer on the robot through the router on the robot.

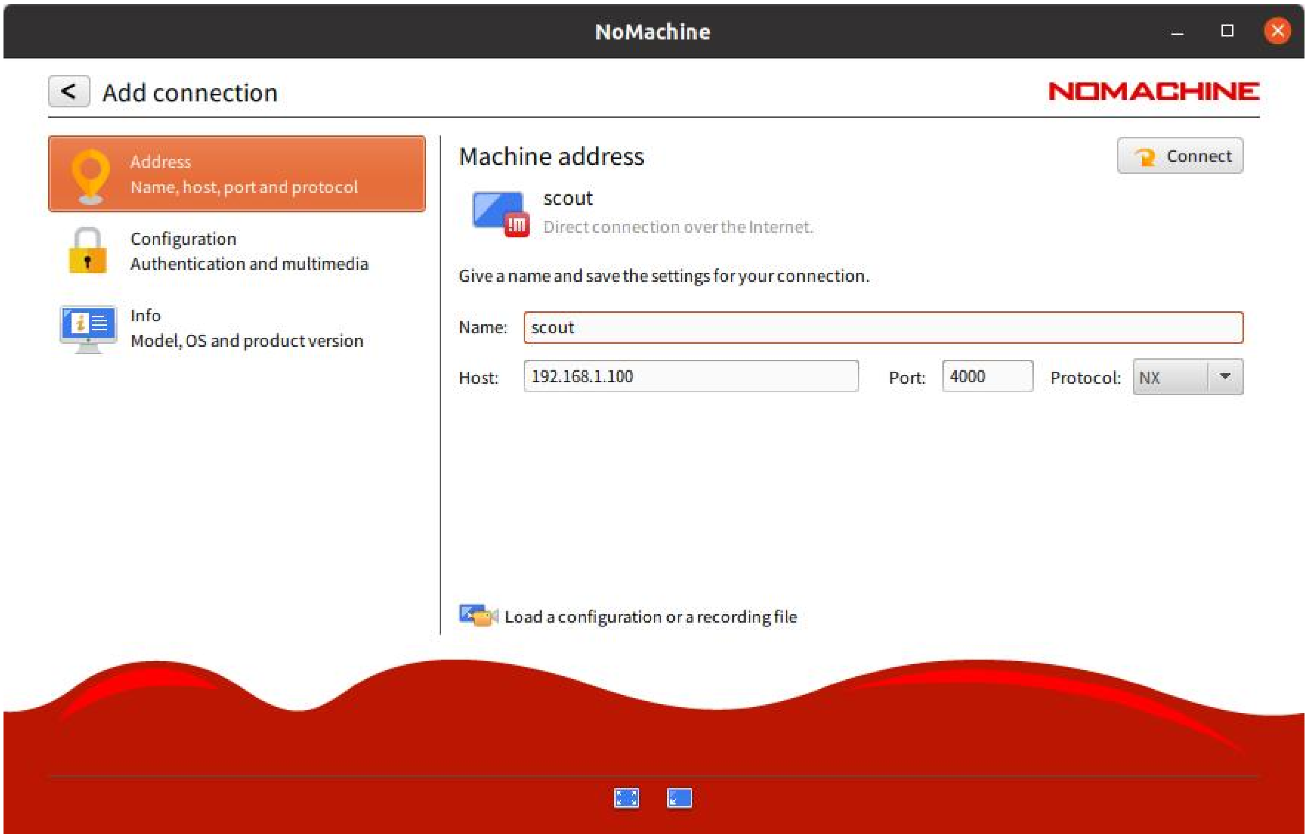
(1) Download the installation package

Choose the corresponding operating system for your laptop or computer to download and install.

(2) How to use it

For users on the industrial computer: the username is "scout" and the startup password for the computer is "agx". The name of the router is "HAIWEI_B316_BD96", and the password for the router login is "12345678", which is the same as the router password. The IP address of the router is 192.168.1.1, and the IP address of the industrial computer is 192.168.1.100.

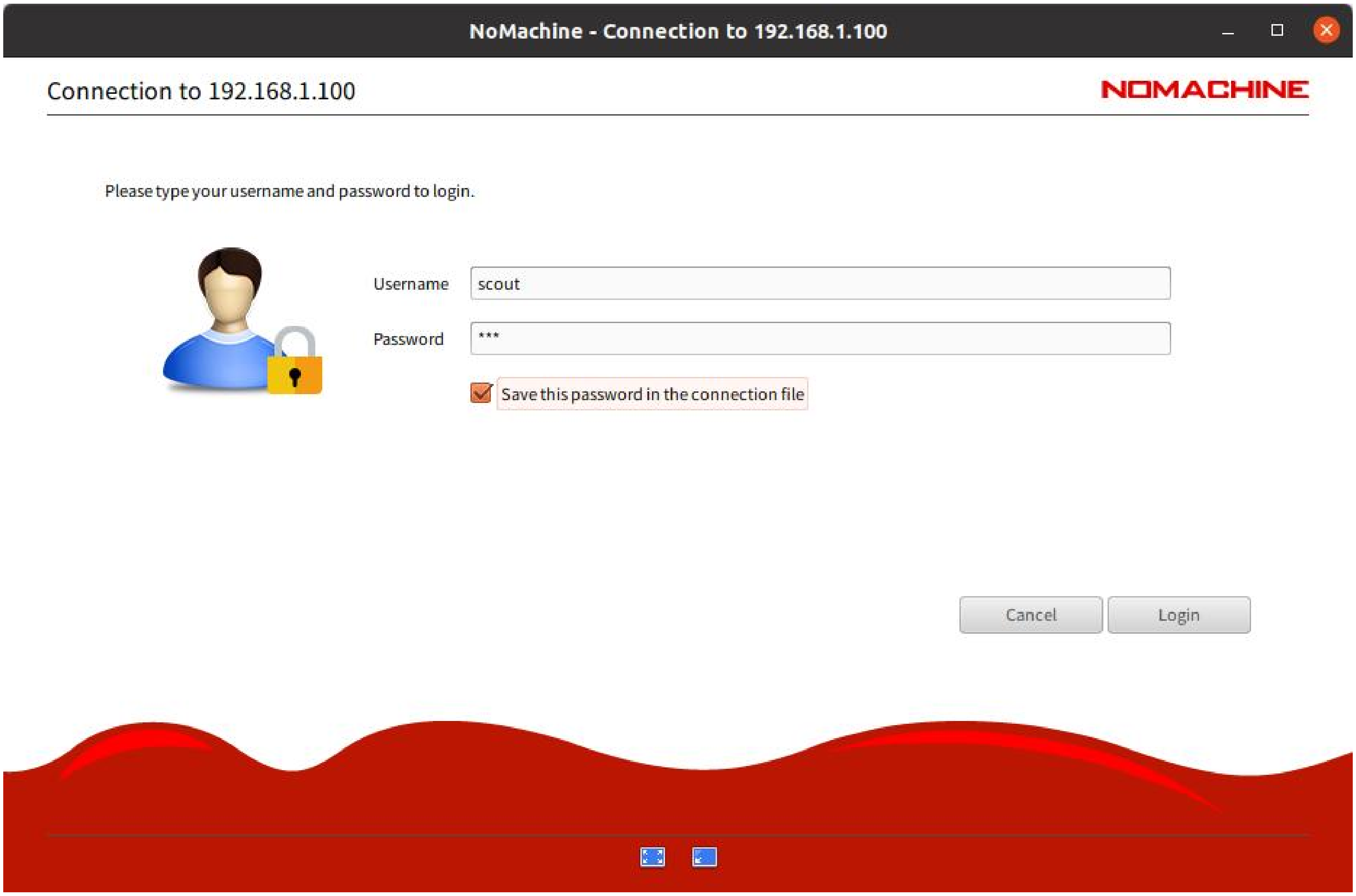
First, use your laptop or computer to find the wifi on the robot and enter the password to connect. Then, open the downloaded remote connection tool NoMachine and add a remote connection.



name：scout，host：192.168.1.100

输入username：scout和password：agx，勾选记住密码。

Enter username: scout and password: agx, check "Remember Password".


NoMachine - Connection to 192.168.1.100
Connection to 192.168.1.100
NOMACHINE
Please type your username and password to login.
Username
scout
Password
***
Save this password in the connection file
Cancel
Login

然后就可以连接到机器人上的工控机了。

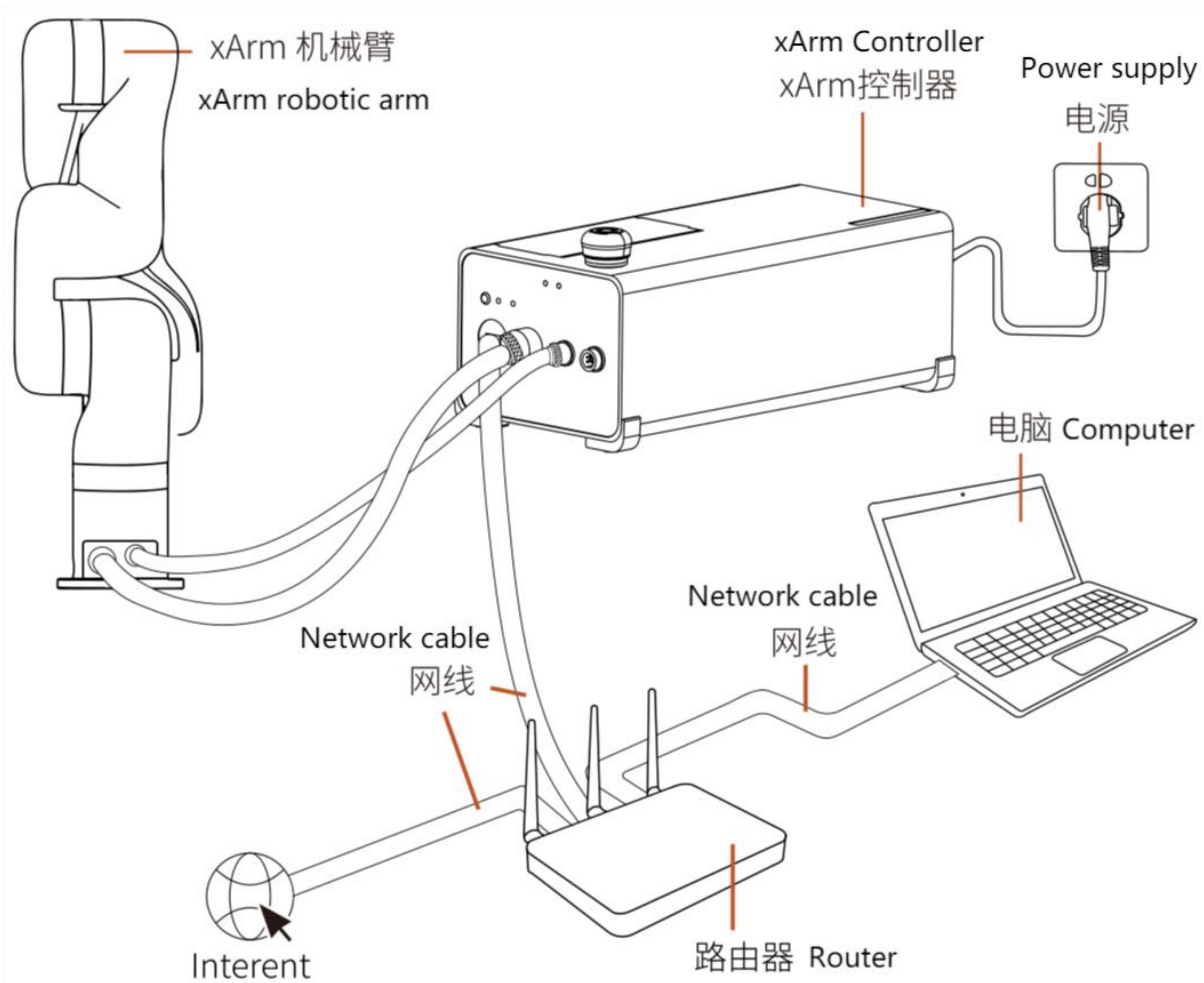
Then it can be connected to the Industrial Personal Computer on the robot.

4.1.2 连接控制机械臂运动

机器人上搭载的机械臂为UFACTORY的xArm 6轴机械臂，机械臂本体、机械臂控制器、机器人底盘控制器（工控机）之间的连接方式如下图所示：

3.1.2 Connecting to control the robot arm movement

The robot is equipped with a 6-axis xArm robotic arm from UFACTORY, and the connection between the robotic arm body, the robotic arm controller, and the robot chassis controller (Industrial Personal Computer) is shown in the following figure:



图中的电脑代表机器人底盘控制器。机械臂控制器与底盘控制器之间通过路由器进行网络通信，局域网的网段为192.168.1，机械臂控制柜傍边有出厂默认IP，这里默认IP地址为192.168.1.198。

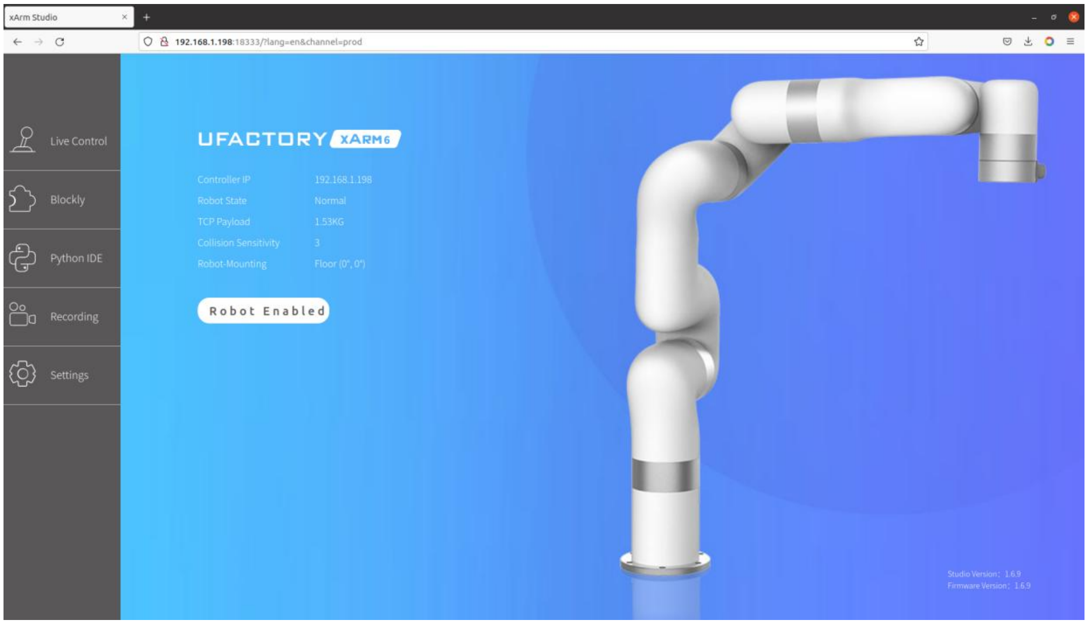
The computer in the figure represents the robot base controller. The communication between the robotic arm controller and the base controller is achieved through a router on the local area network with the subnet of 192.168.1. There is a factory default IP next to the control cabinet of the robotic arm, here default IP address of 192.168.1.198.

- **界面控制**

Interface control

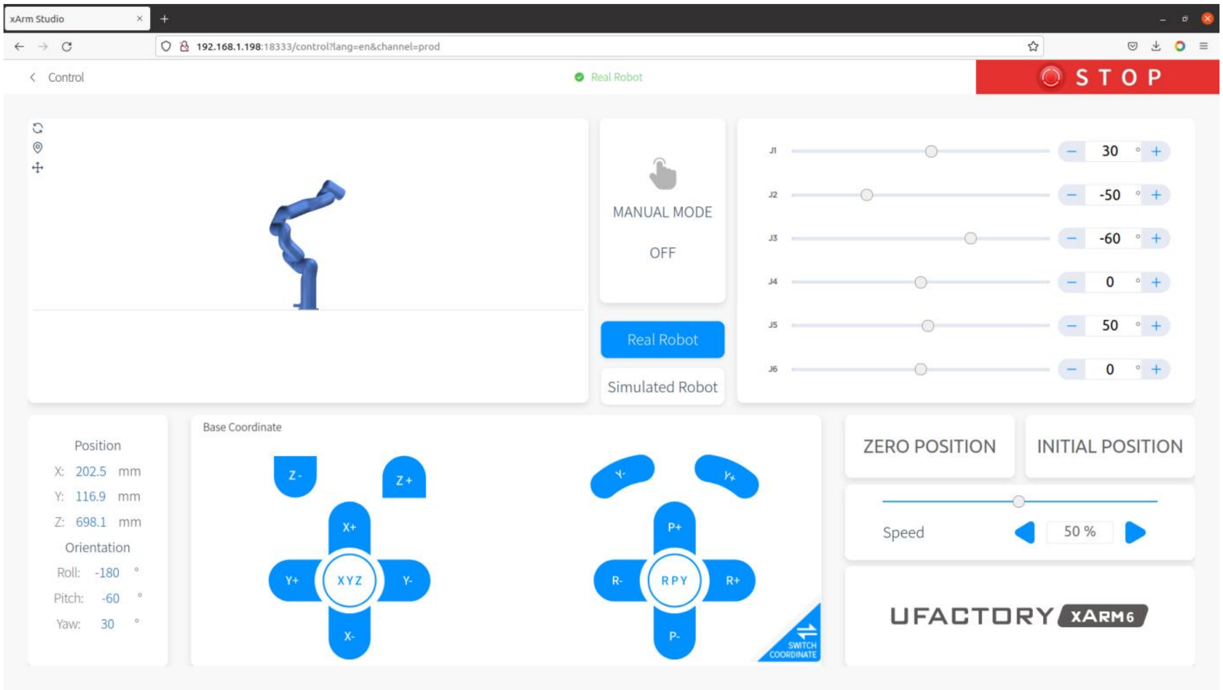
打开机器人电源后，我们打开浏览器，在地址栏输入192.168.1.198:18333，回车进入机械臂Web界面：

After powering on the robot, open a web browser and enter 192.168.1.198:18333 in the address bar to access the xArm web interface.



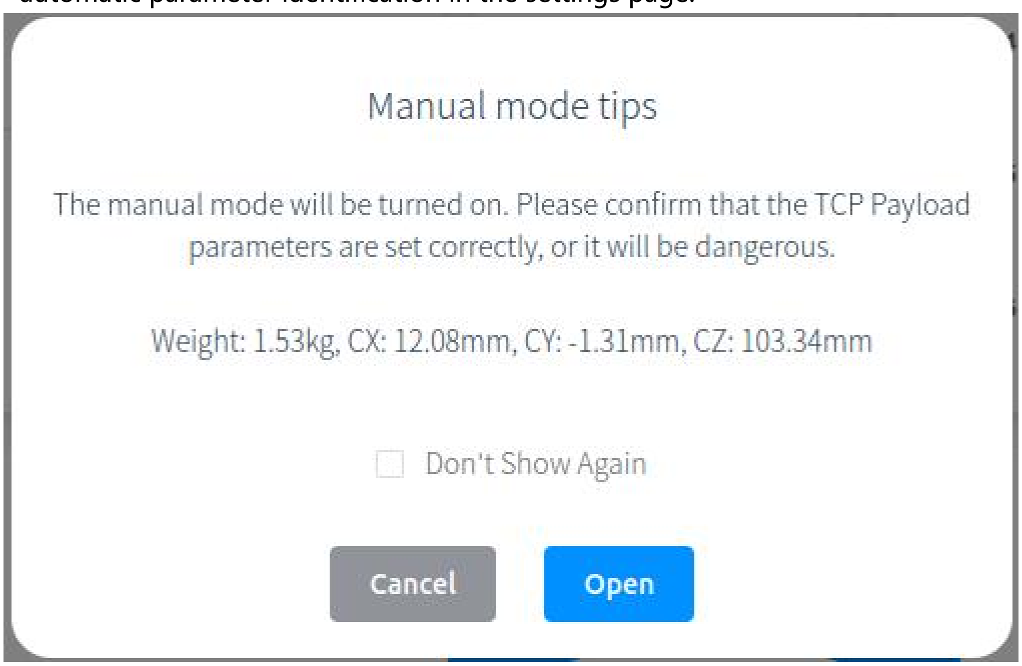
点击左侧的Live　Control，进入如下图所示的机械臂实时控制界面。这里我们可以看到机械臂的状态位形、机械臂的末端位置和姿态以及每个关节的角度数据，同时我们也可以通过相应的按钮控制机械臂的每个关节进行运动，或者直接控制机械臂末端在基坐标系下的位姿。

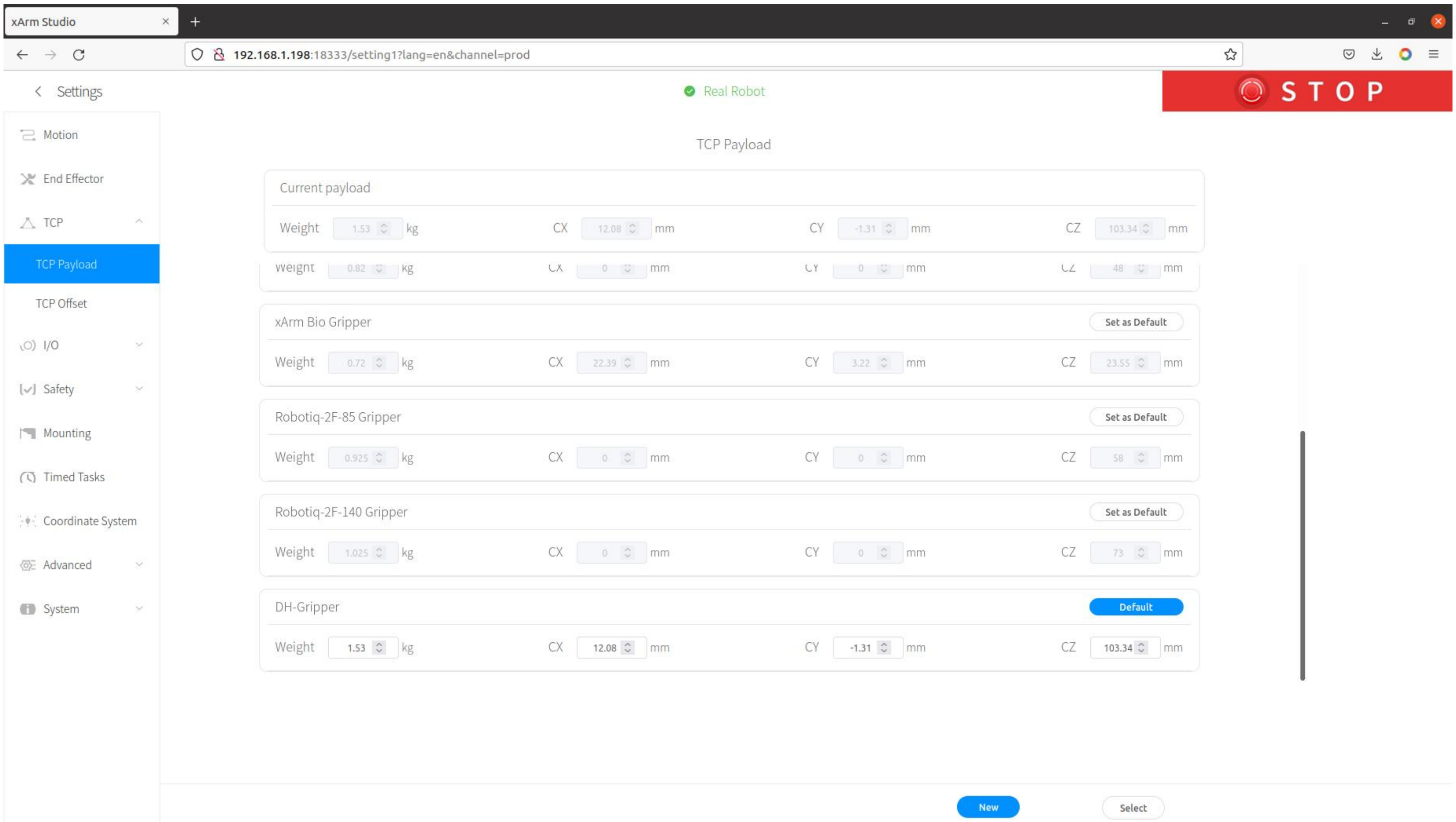
Click on the Live Control button on the left, enter the real-time control interface of the robot arm as shown in the figure below. Here, you can see the status, end position and posture of the robot arm, as well as the angle data of each joint. At the same time, we can control the movement of each joint of the robot arm through the corresponding buttons, or directly control the pose of the end effector of the robot arm in the base coordinate system.



在界面中，我们可以点击MANUAL MODE按钮，使其变为ON启动手动模式，此时我们便可以来拖动机械臂运动。 注意：开启手动模式之前，要配置机械臂末端工具的负载参数，默认已经配置好，用户也可以在设置页面中进行自动参数辨识。

In the interface, we can click the "MANUAL MODE" button to turn it ON and start the manual mode. Then, we can drag the robot arm to move. Note: Before starting the manual mode, the payload parameters of the end of robotic arm tooling should be configured. The default settings are already configured, and users can also perform automatic parameter identification in the settings page.





上图TCP Payload中的DH Gripper参数为自适应辨识参数，它包括了机械臂末端安装的大寰机械手爪和RealSense相机。

-

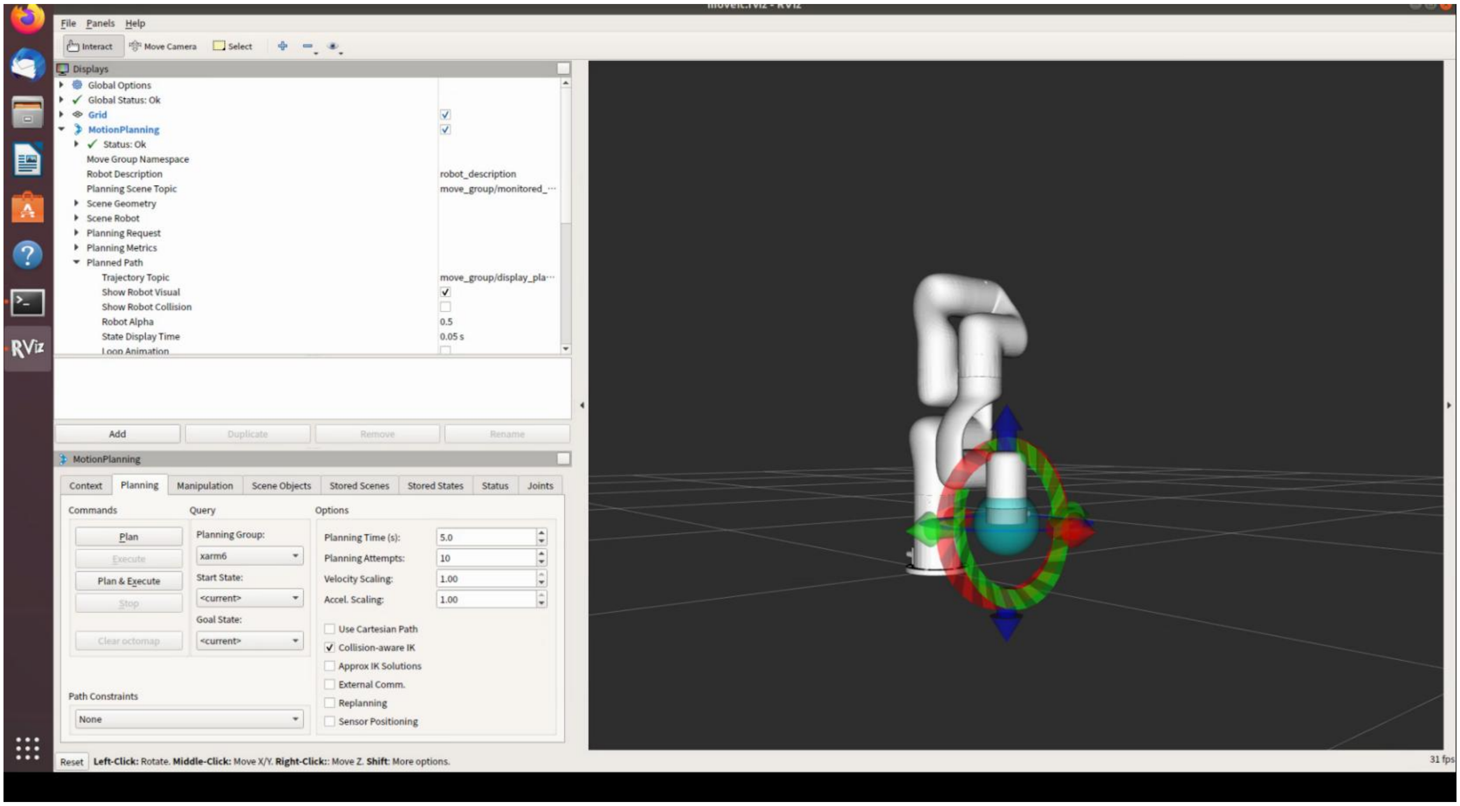
The DH Gripper parameter in the TCP Payload above is an adaptive identification parameter, which includes the DH robotic gripper and RealSense camera installed at the end of the robotic arm.

MoveIt!控制

MoveIt! control

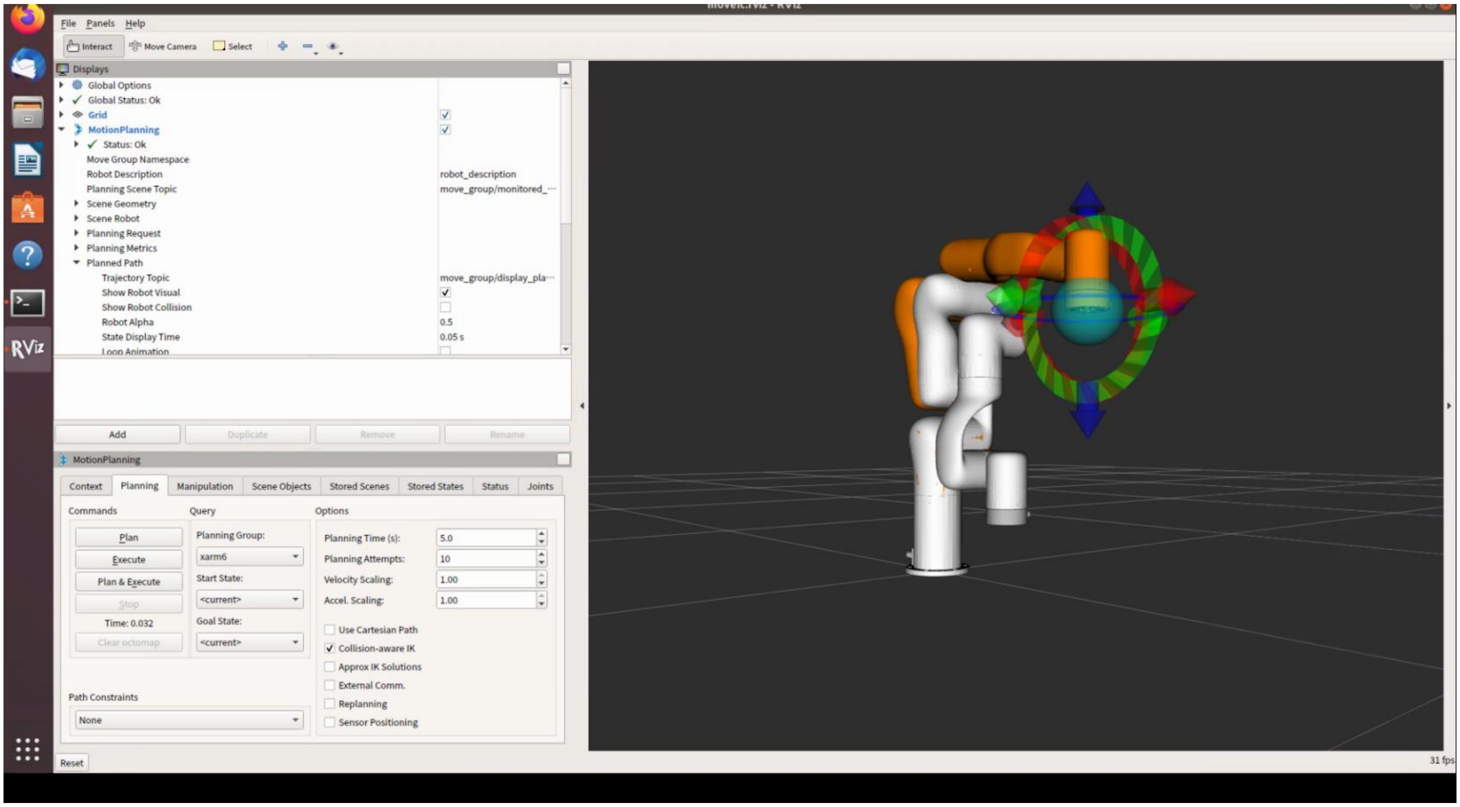
在机器人中打开终端，运行命令 `roslaunch xarm6_moveit_config realMove_exec.launch robot_ip:=192.168.1.198` ，启动MoveIt!机械臂规划控制。

Open the terminal in the robot and run the command "roslaunch xarm6_moveit_config realMove_exec.launch robot_ip:=192.168.1.198" to start MoveIt! Robotic arm planning control.



我们可以用鼠标拖动RViz中机械臂的末端到另一个位姿，然后点击左下角MotionPlanning中的Plan按钮，看MoveIt是否可以从当前位姿规划到目标位姿，如果路径规划成功且路径过程中没有其他障碍物，则点击Plan按钮控制机械臂运动到目标位姿。

We can drag the end effector of the robotic arm in RViz to another pose using the mouse. Then, click on the "Plan" button in the "MotionPlanning" section at the bottom left corner of the screen to see if MoveIt can plan a path from the current pose to the target pose. If the path planning is successful and there are no obstacles in the way, click the "Plan" button again to control the robotic arm to move to the target pose.



- **ROS命令控制**

ROS command control

机械臂有三种常用的控制模式，分别为：

Mode 0：基于xArm controller规划的位置模式；

Mode 1：基于外部轨迹规划器的位置模式；

Mode 2：自由拖动(零重力)模式。

MoveIt!控制时使用的是Mode1，在使用ROS命令控制之前需要先将模式切换到Mode 0：

There are three commonly used control modes for the robotic arm, which are as follows:
Mode 0: Position mode based on the xArm controller's planning;
Mode 1: Position mode based on an external trajectory planner;
Mode 2: Free dragging (zero gravity) mode.
When using MoveIt! control, Mode 1 is used. Before using ROS command control, switch the mode to Mode 0.

```
$ rosservice call /xarm/clear_err        #清除错误，发生错误时运行
$ rosservice call /xarm/motion_ctrl 8 1    #关节使能，关节未使能时运行
$ rosservice call /xarm/set_mode 0         #设置期望的运行模式
$ rosservice call /xarm/set_state 0       #set state 为0（Ready状态）
```

#Clear errors; run if an error occurs

#Joint enables, run if joints are not enabled

#Set the desired operating mode

#Set state to 0 (Ready state)

模式切换完成后，可通过如下命令控制机械臂在关节空间运动：

After completing the mode switch, the robotic arm can be controlled to move in joint space using the following command:

```
$ rosservice call /xarm/move_joint [0.5236,0,0,0,-1.5708,0] 0.5 5 0 0
```

通过如下命令控制机械臂在笛卡尔空间运动:

The robotic arm can be controlled to move in Cartesian space using the following command:

```
$ rosservice call /xarm/move_line "pose: [300, 200, 300, 3.14, 0, 0.55]
mvvelo: 50.0
mvacc: 500.0
mvtime: 0.0
mvradii: 0.0"
```

通过如下命令使机械臂归零:

Reset the robotic arm using the following command:

```
$ rosservice call /xarm/go_home "pose: [0]
mvvelo: 0.5
mvacc: 5.0
mvtime: 0.0
mvradii: 0.0"
```

```
scout@agilex:~$ rosservice call /xarm/move_joint [0.5236,0,0,0,-1.5708,0] 0.5 5 0 0
ret: 0
message: "move joint, ret = 0"
scout@agilex:~$ rosservice call /xarm/move_line "pose: [300, 200, 400, 3.14, 0, 0.55]
mvvelo: 50.0
mvacc: 500.0
mvtime: 0.0
mvradii: 0.0"
ret: 0
message: "move line, ret = 0"
scout@agilex:~$ rosservice call /xarm/go_home "pose: [0]
mvvelo: 0.5
mvacc: 5.0
mvtime: 0.0
mvradii: 0.0"
ret: 0
message: "go home, ret = 0"
```

4.1.3 使用ar_track_alvar实现视觉定位追踪

4.1.3.1 功能简介

这个包是一个开源 AR 标签跟踪库 Alvar 的 ROS PACKAGE。

（1） ar_track_alvar 有 4 个主要功能： 生成不同大小、分辨率和数据/ID 编码的 AR 标签

（2） 识别和跟踪单个 AR 标签的姿势，可选择集成深度相机的深度数据以获得更好的姿势估计。

（3）识别和跟踪由多个标签组成的组合姿势。这允许更稳定的姿态估计、对遮挡的鲁棒性以及对多边对象的跟踪。

（4）使用相机图像自动计算捆绑中标签之间的空间关系，这样用户就不必手动测量并在 XML 文件中输入标签位置来使用捆绑功能（目前不工作）。

Alvar 比 ARToolkit 更新、更先进，ARToolkit 一直是其他几个 ROS AR 标签包的基础。 Alvar 具有自适应阈值处理以处理各种光照条件、基于光流的跟踪以实现更稳定的姿态估计，以及改进的标签识别方法，该方法不会随着标签数量的增加而 显着减

慢。

3.1.3 Using ar_track_alvar for Visual Localization and Tracking

3.1.3.1 Functionality Overview

This package is a ROS package for the open-source AR tag tracking library Alvar.

ar_track_alvar has four main functionalities:

(1) Generate AR tags with different sizes, resolutions, and data/ID encoding.

(2) Recognize and track the pose of a single AR tag, optionally integrating depth data from a depth camera for better pose estimation.

(3) Recognize and track the combined pose of multiple tags. This allows for more stable pose estimation, robustness against occlusions, and tracking of multi-sided objects.

(4) Automatically compute the spatial relationships between tags in a bundle using camera images, so that users do not have to manually measure and input tag positions in an XML file to use the bundle functionality (currently not working).

Compared to ARToolkit, Alvar is more updated and advanced. ARToolkit has been the foundation of several other ROS AR tag packages. Alvar features adaptive thresholding to handle various lighting conditions, optical flow-based tracking for more stable pose estimation, and an improved tag recognition method that does not significantly slow down with increasing tag numbers.

4.1.3.2 使用说明

3.1.3.2 Instructions for use

（1）生成标签

Generate tags

(1)

```
$ source ~/catkint_workspace/devel/setup.bash
$ rosrun ar_track_alvar createMarker 0 -s 3.0
```

> 后面的数字表示生成标签的id号，可以根据需要生成不同数字的标签

The numbers following indicate the ID number for

generating tags, which can be adjusted according to the needs

of different tag numbers.

详细的参数设置如下图：

Detailed parameter settings are shown in the figure below:

```
65535               marker with number 65535
-f 65535            force hamming(8,4) encoding
-1 "hello world"    marker with string
-2 catalog.xml      marker with file reference
-3 www.vtt.fi       marker with URL
-u 96               use units corresponding to 1.0 unit per 96 pixels
-uin                use inches as units (assuming 96 dpi)
-ucm                use cm's as units (assuming 96 dpi) <default>
-s 5.0              use marker size 5.0x5.0 units (default 9.0x9.0)
-r 5                marker content resolution -- 0 uses default
-m 2.0              marker margin resolution -- 0 uses default
-a                  use ArToolkit style matrix markers
-p                  prompt marker placements interactively from the user
```

（2）打印标签

打印标签时注意尺寸，默认使用3x3 cm的标签，如果打印尺寸和默认尺寸不同，请在~/your_workspace/src/open_manipulator_perceptions/open_manipulator_ar_markers/launch/agx_ar_pose.launch文件中修改标签尺寸。

(2) Printing tags

When printing tags, pay attention to the size. The default size is 3x3 cm. If the printing size is different from the default size, please modify the tag size in the ~/your_workspace/src/open_manipulator_perceptions/open_manipulator_ar_markers/launch/agx_ar_pose.launch file.

```
<arg name="user_marker_size"     default="3"/>
```

> 在使用过程中默认0号标签贴在物体上，2号标签贴在物品摆放平台上。
>
> In the process of use, the default tag 0 is attached to the object and tag 2 is attached to the object placement platform.

（3）运行识别功能

用usb3.0的线连接摄像头与电脑，运行下面指令：

(3) Running recognition function

Connect the camera to the computer using a USB3.0 cable and run the following command:

```
$ source ~/catkin_workspace/devel/setup.bash
$ roslaunch agx_xarm_bringup agx_ar_pose.launch
```

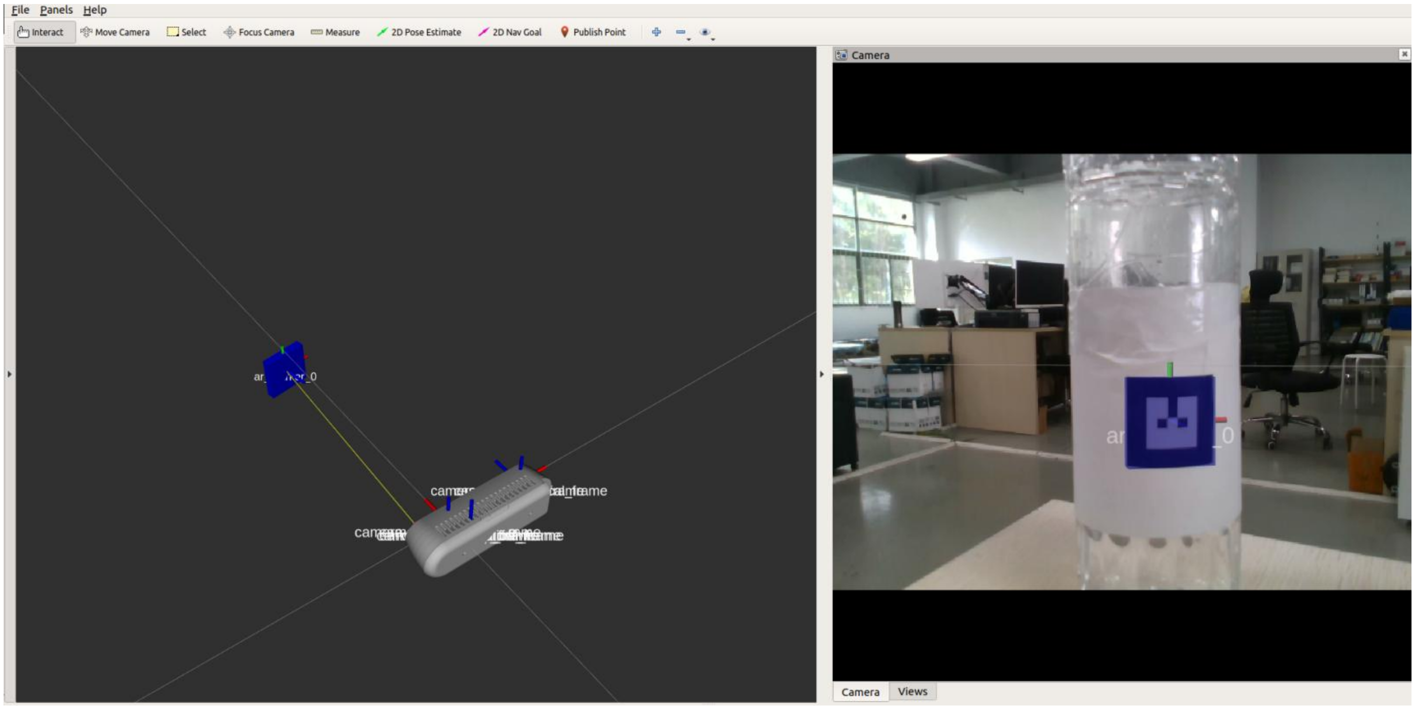
其agx_ar_pose.launch文件的结构如下图所示，详细参数参照下表：

The structure of the agx_ar_pose.launch file is shown in the figure below, with detailed parameters referenced in the table below:

```
<?xml version="1.0" ?>
<launch>
  <arg name="x"              default="0"/>
  <arg name="y"              default="0"/>
  <arg name="z"              default="0"/>
  <arg name="roll"              default="0"/>
  <arg name="pitch"              default="0"/>
  <arg name="yaw"              default="0"/>
  <arg name="r_x"              default="0"/>
  <arg name="r_y"              default="0"/>
  <arg name="r_z"              default="0"/>
  <arg name="r_w"              default="0"/>
  <arg name="parent_link"              default="wrist3_Link"/>
  <arg name="serial_no"              default=""/>
  <arg name="marker_frame_id"     default="_color_frame"/>
  <arg name="user_marker_size"    default="3"/>
  <arg name="use_quaternion" default="false"/>
  <arg name="camera_model" default="realsense_d435" doc="model type [astra_pro, realsense_d435, raspicam]"/>
  <arg name="camera_namespace" default="camera"/>

  <group if="$(eval camera_model == 'realsense_d435')">
    <include file="$(find realsense2_camera)/launch/rs_camera.launch">
      <arg name="camera"              value="$(arg camera_namespace)"/>
      <arg name="enable_pointcloud"   value="false" />
      <arg name="serial_no"           value="$(arg serial_no)"/>
    </include>

    <node unless="$(arg use_quaternion)" pkg="tf" type="static_transform_publisher" name="$(arg camera_namespace)_to_realsense_frame"
      args="$(arg x) $(arg y) $(arg z) $(arg yaw) $(arg pitch) $(arg roll) $(arg parent_link) $(arg camera_namespace)_link 10" />

    <node if="$(arg use_quaternion)" pkg="tf" type="static_transform_publisher" name="$(arg camera_namespace)_to_realsense_frame"
      args="$(arg x) $(arg y) $(arg z) $(arg r_x) $(arg r_y) $(arg r_z) $(arg r_w) $(arg parent_link) $(arg camera_namespace)_link 10" />

    <include file="$(find ar_track_alvar)/launch/pr2_indiv_no_kinect.launch">
      <arg name="marker_size" value="$(arg user_marker_size)" />
      <arg name="max_new_marker_error" value="0.08" />
      <arg name="max_track_error" value="0.2" />
      <arg name="cam_image_topic" value="$(arg camera_namespace)/color/image_raw" />
      <arg name="cam_info_topic" value="$(arg camera_namespace)/color/camera_info" />
      <arg name="output_frame" value="$(arg camera_namespace)$(arg marker_frame_id)" />
      <arg name="node_name" value="$(arg camera_namespace)"/>
    </include>
  </group>

</launch>
```

参数 parameter	默认值 default	功能 function

x,y,z	0	摄像头固定在机械臂中的静态位置 The static position of the camera fixed in the robot arm
roll,pitch,yaw	0	摄像头固定在机械臂中的静态姿态欧拉角表示形式 The static orientation of the camera fixed in the robot arm represented in Euler angles
r_w,r_x,r_y,r_z	0	摄像头固定在机械臂中的静态姿态四元数表示形式 The static orientation of the camera fixed in the robot arm represented in quaternion form
parent_link	wrist3_Link	摄像头发布tf的相对坐标系 The relative coordinate system of the camera published through tf
serial_no	空 null	启动摄像头的序列号 The serial number to activate the camera
user_marker_size	3	标签的尺寸大小（cm） The size of the tag (in cm) used for detection
use_quaternion	false	是否用四元数表示姿态 hould the orientation be represented using quaternions
camera_model	realsense_d435	摄像头模型 The camera mode
camera_namespace	camera	摄像头命名空间 The camera namespace



4.1.4 使用Gmapping构建地图

4.1.4.1 功能简介

gmapping功能包订阅机器人的深度信息、IMU信息和里程计信息，同时完成一些必要参数的配置，即可创建并输出基于概率的二维栅格地图。gmapping功能包基于openslam社区的开源SLAM算法。

3.1.4 Using Gmapping to build a map

3.1.4.1 Function overview

The gmapping package subscribes to the robot's depth information, IMU information, and odometry information, and completes the necessary parameter configuration to create and output a probability-based two-dimensional grid map. The gmapping package is based on the open-source SLAM algorithm from the open-source slam community.

4.1.4.2 功能运行

3.1.4.2 Function operation

```
$ roslaunch agilexpro open_lidar.launch
$ roslaunch agilexpro gmapping.launch
```

构建好地图之后，把地图保存在~/catkin_workspace/src/agilexpro/maps目录下

After the map is built, save it in the ~/catkin_workspace/src/agilexpro/maps directory.

```
$ roscd agilexpro/maps
$ rosrun map_server map_saver -f map
```

> 最后面的map是生成地图的名字，在导航的时候默认是加载以map命名地图文件，如果在命名的时候取了别的名字需要修改加载地图的名字。把~catkin_workspace/src/agilexpro/launch/navigation_4wd.launch文件中下面地图名字改为你命名的名字。
>
> The last "map" is the name of the generated map. When navigating, the default is to load the map file named "map". If you named it something else, you need to modify the name of the loaded map in the ~/catkin_workspace/src/agilexpro/launch/navigation_4wd.launch file.

```
<node name="map_server" pkg="map_server" type="map_server" args="$(find agilexpro)/maps/map.yaml" output="screen">
```

4.1.5 使用move_base导航

4.1.5.1 功能简介

move_base 包提供了一个动作的实现（参见 actionlib 包），给定世界上的目标，它将尝试使用移动基地来实现它。move_base 节点将全局和本地规划器链接在一起以完成其全局导航任务。move_base 节点还维护两个成本地图，一个用于全局规划器，另一个用于本地规划器（参见 costmap_2d 包），用于完成导航任务。

3.1.5 Using move_base navigation

3.1.5.1 Function overview

The move_base package provides an implementation of an action (see the actionlib package). Given a goal in the world, it will attempt to reach it using the mobile base. The move_base node links the global and local planners together to accomplish its global navigation task. The move_base node also maintains two cost maps, one for the global planner and one for the local planner (see the costmap_2d package), which are used to accomplish the navigation task.

4.1.5.2 功能运行

3.1.5.2 Function operation

```
$ roslaunch agilexpro open_lidar.launch
$ roslaunch agilexpro navigation.launch
```

4.1.6 使用smach库实现任务状态切换

3.1.6.1功能简介

SMACH中明确描述所有可能的状态和状态转换，在机器人执行一些复杂的计划时，很有用。 这基本上消除了将不同模块组合在一起的冲突，使移动机器人操控等系统做更多有趣的事情

3.1.6 Using the SMACH library to implement task state switching

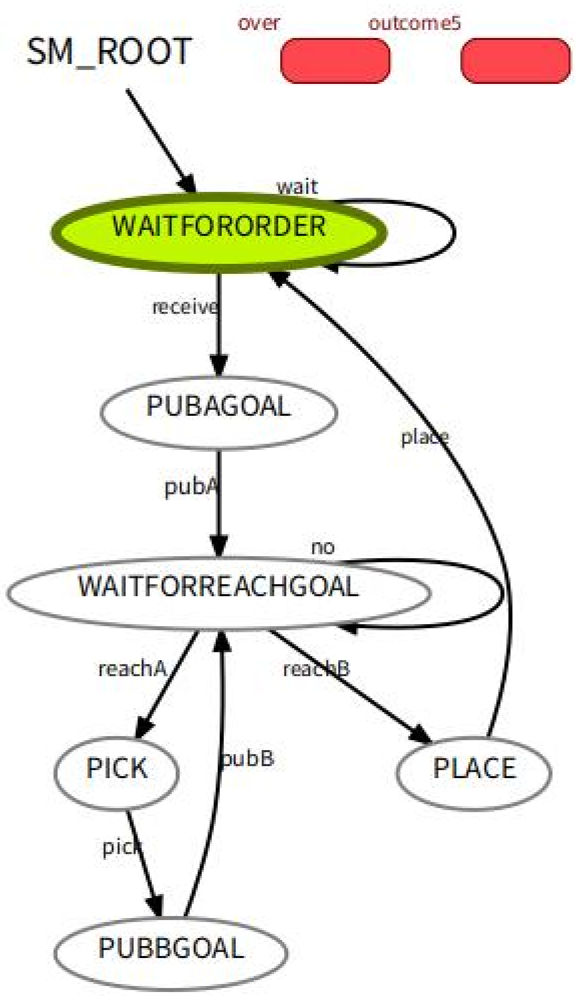
3.1.6.1 Function overview

SMACH explicitly describes all possible states and state transitions, which is useful when a robot performs complex plans. This essentially eliminates conflicts in combining different modules, enabling mobile robot control systems to do more interesting things.

3.1.6.2功能运行

3.1.6.2 Function operation

```
$ source ~/catkin_workspace/devel/setup.bash
$ rosrun agx_xarm_smach smach_demo.py
$ rosrun smach_viewer smach_viewer.py
```



4.1.7 使用MoveIt对机械臂进行路径规划

4.1.7.1 功能简介

利用MoveIt成熟的运动规划器对机械臂的运动进行路径规划、碰撞检测和路径执行。从而在给定目标点的时候可以计算出一条最优路径出来。

3.1.7 Using MoveIt for robot arm path planning

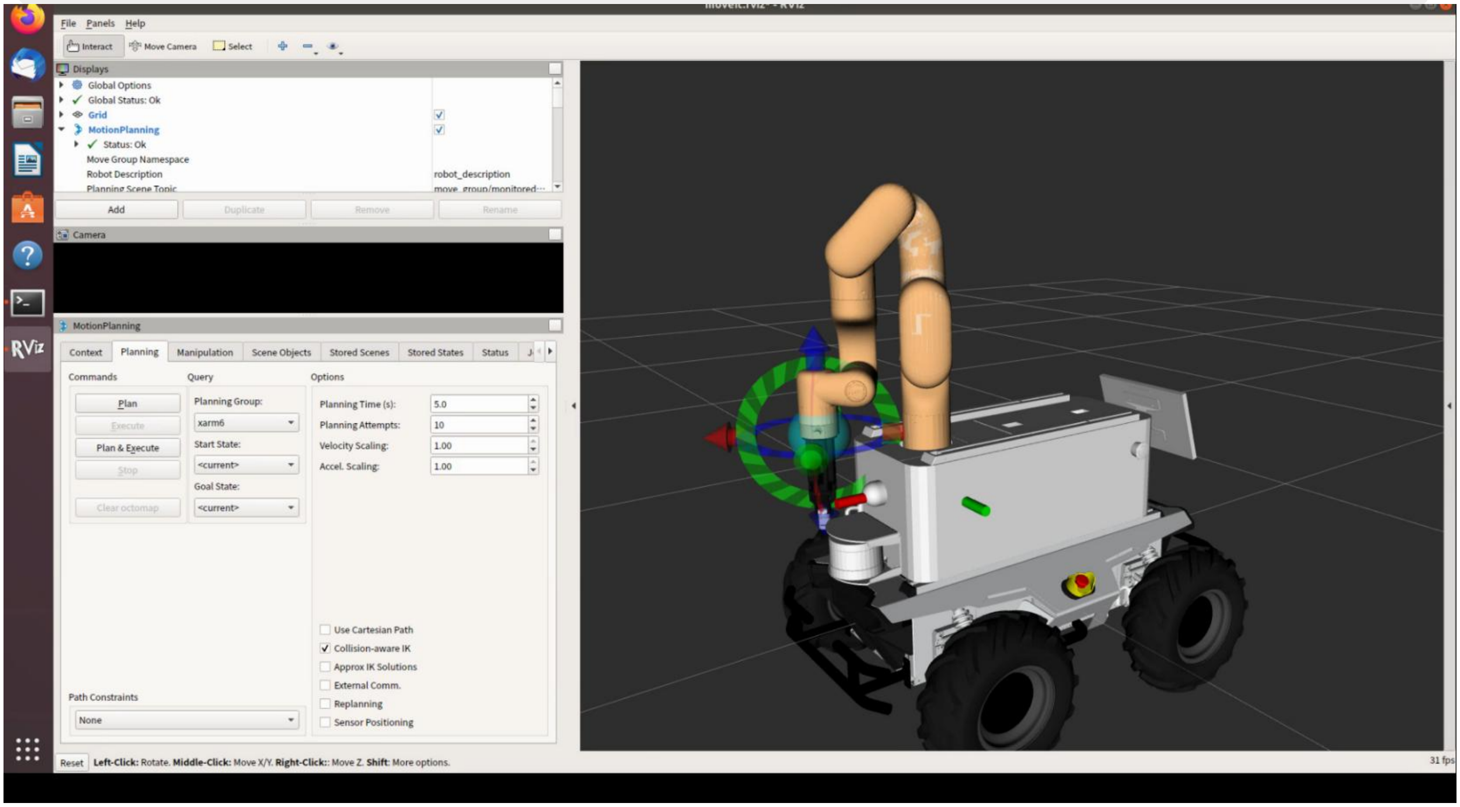
3.1.7.1 Function introduction

Using the mature motion planner provided by MoveIt, the robot arm's motion is planned, collision detection is performed, and the optimal path is calculated when a target point is given.

4.1.7.2 功能运行

3.1.7.2 Function operation

```
$ roslaunch agx_xarm_bringup setup_arm.launch
```



这是完整的底盘与机械臂的组合模型，我们可以拖动机械臂末端执行路径规划，此时的机械臂不会与底盘发生碰撞。

This is the complete model of the chassis and the robot arm. We can drag the end of the robot arm to perform path planning, and the robot arm will not collide with the chassis.

4.2 功能使用

4.2.1 启动机械臂

3.2 Function usage

3.2.1 Start the robot arm

```
$ roslaunch agx_xarm_bringup setup_arm.launch
```

> 开启机械臂控制柜的电源，解锁机械臂每个关节的保险。长按机械臂控制柜上连接的示教显示屏上的电源按钮，在弹出的小窗口点击保存->启动，就开启了机械臂的电气开关。

Turn on the power of the robot arm control cabinet, unlock the insurance of each joint of the robot arm. Press and hold the power button on the teaching display screen connected to the robot arm control cabinet, and click "Save"->"Start" in the pop-up window to turn on the electrical switch of the robot arm.

在这个launch file里面可支持修改一个参数，robot_ip。这里传入的ip为机械臂控制柜默认ip：192.168.1.198。

One parameter robot_ip, can be modified in this launch file. The IP passed in here is the default IP of the robot arm control cabinet: 192.168.1.198.

```
<launch>
  <arg name="robot_ip" default="192.168.1.198" />
  <include file="$(find scout_xarm_moveit_config)/launch/realMove_exec.launch">
    <arg name="robot_ip" value="$(arg robot_ip)"/>
  </include>
  <node pkg="dh_gripper_driver" type="dh_gripper_joint_state" name="dh_gripper_state" />
</launch>
```

4.2.2 启动摄像头

3.2.2 Start the camera

```
$ roslaunch agx_xarm_bringup open_camera.launch
```

> 注意启动之前需要用USB3.0的数据线连接摄像头与工控机。

Before starting, connect the camera to the IPC with a

USB3.0 data cable.

```
  <arg name="camera_namespace" value="cam1"/>
<arg name="serial_no" value="035422071503"/>
<arg name="node_name" value="cam1"/>
  <arg name="use_quaternion" value="true"/>
<!--<arg name="roll" value="0"/>
<arg name="pitch" value="-1.57"/>
<arg name="yaw" value="-1.57"/>
<arg name="x"  value="0.05"/>
<arg name="y"  value="0.03"/>
<arg name="z" value="0.0"/>-->
<arg name="x" value="0.0446414171704"/>
<arg name="y" value="0.053996778351"/>
<arg name="z" value="0.0600140964856"/>
<arg name="r_x" value="-0.000283027265062"/>
<arg name="r_y" value="-0.0682970373998"/>
<arg name="r_z" value="0.984025866108"/>
<arg name="r_w" value="0.164403556559"/>
```

该launch file支持修改的参数有下面几个：

- camera_namespace：摄像头节点的命名空间，当启动多个摄像头时可以需要修改每个摄像头的命名空间。
- serial_no：摄像头的序列号S/N码，当需要启动多个摄像头的时候可以传入摄像头的序列号指定打开某个摄像头。
- node_name：一般与命名空间相同。
- use_quaternion：true表示使用四元数，false表示使用欧拉角。
- x,y,z：摄像头距离机械臂的位置。
- roll,pitch,yaw：摄像头姿态欧拉角。
- r_x,r_y,r_z,r_w：摄像头姿态四元数。

Several parameters can be modified in this launch file:

- camera_namespace: the namespace of the camera node. When multiple cameras are started, the namespace of each camera may need to be modified.
- serial_no: the serial number of the camera,S/N code. When multiple cameras need to be started, the serial number of the camera can be passed in to specify which camera to open.
- node_name: generally the same as the namespace.
- use_quaternion: true for using quaternions, false for using Euler angles.
- x,y,z: the position of the camera relative to the robot arm.
- roll,pitch,yaw: the attitude Euler angles of the camera.
- r_x,r_y,r_z,r_w: the attitude quaternion of the camera.

 在启动这个launch file的时候也启动了图像识别节点，会实时更新物体和物体摆放平台的空间坐标信息，可根据以下接口获取这两点信息。服务名字为 /pick_point。服务类型为agx_pick_msg/AgxPickSrv ，详细的消息格式如下：

When starting this launch file, the image recognition node is also started, which will update the spatial coordinate information of the object and the object placement platform in real-time. This information can be obtained through the following interface. The service name is /pick_point. The service type is agx_pick_msg/AgxPickSrv, and the detailed message format is as follows:

```
int32 item          # 0--bottle 1--pick 2--place
int32 handpose      # 夹爪夹取物体的方向 0--front 1--upside
int32 Prepose       # 预备夹取点.0--物体真实位置；
                    # 1--根据handpose往前或者往上偏移一个夹爪的位置
---
#返回值
geometry_msgs/Point position
  float64 x
  float64 y
  float64 z
geometry_msgs/Quaternion orientation
  float64 x
  float64  y
  float64  z
  float64 w
```

#The direction of the gripper to grasp the object. 0--front 1--upside

#The preparatory grasping point. 0--the real position of the object;

#1-- According to the handpose offset from the hand pose forward or upward by one gripper position

其中q为0时，返回物体的坐标信息；q为2时，返回物体摆放位置信息。

When q is 0, the coordinate information of the object is returned; when q is 2, the information of the object placement position is returned.

4.2.3 启动机械爪

3.2.3 Start the gripper

```
$ roslaunch agx_xarm_bringup open_gripper.launch
```

控制机械爪的topic接口为/gripper/ctrl,消息的类型为dh_gripper_msgs/GripperCtrl ，消息格式如下：

The topic interface for controlling the gripper is /gripper/ctrl, and the message type is dh_gripper_msgs/GripperCtrl. The message format is as follows:

```
bool initialize        #是否初始化
float32 position       #两个夹片的间距
float32 force          #夹取物体的力度
float32 speed          #二指夹爪这个参数无效，夹取速度和force正相关
```

#Whether to initialize

#The distance between the two gripper pieces

#The force used to grasp the object

#For the two-fingered gripper, the parameter has no effect, and the grasping speed and force are positively correlated

在agx_xarm_pick功能包里面有个一demo文件可以输入指尖位置来控制机械爪。使用指令如下：

In the agx_xarm_pick package, there is a demo file that can control the gripper by inputting the position of the fingertip. The command is as follows:

```
$ rosrun agx_xarm_pick control_gripper #q退出
```

q to exit

4.2.4 启动agx_xarm_pick节点

3.2.4 Start the agx_xarm_pick node

```
$ roslaunch agx_xarm_bringup agx_xarm_bring.launch
```

该节点需要获取状态机传入的taskid来分别控制机器人执行不同的任务。任务的服务接口为/send_task，服务类型为agx_pick_msg/TaskCmd，消息格式如下：

The node needs to obtain the task ID passed by the state machine to control the robot to perform different tasks. The service interface for the task is /send_task, with the service type agx_pick_msg/TaskCmd. The message format is as follows:

```
int32 taskID
geometry_msgs/PoseStamped A_goal
```

```
  std_msgs/Header header
    uint32 seq
    time stamp
    string frame_id
  geometry_msgs/Pose pose
    geometry_msgs/Point position
    float64 x
      float64 y
      float64 z
    geometry_msgs/Quaternion orientation
      float64 x
      float64  y
      float64  z
      float64 w
geometry_msgs/PoseStamped   B_goal
  std_msgs/Header header
    uint32 seq
    time stamp
    string frame_id
  geometry_msgs/Pose pose
    geometry_msgs/Point position
    float64 x
      float64 y
      float64 z
    geometry_msgs/Quaternion orientation
      float64 x
      float64  y
      float64  z
      float64 w
---
bool result
```

4.2.5 启动smach状态机节点

3.2.5 Start smach state machine node

```
$ rosrun agx_xarm_smach smach_demo.py
```

定义了以下几种状态的转换：

Transitions for the following states are defined:

```
smach.StateMachine.add('WAITFORORDER', WaitFororder(),
       transitions={'receive':'PUBAGOAL',
   'wait':'WAITFORORDER'})

smach.StateMachine.add('PUBAGOAL', PubAGoal(),
       transitions={'pubA':'WAITFORREACHGOAL'})

smach.StateMachine.add('WAITFORREACHGOAL', WaitForReachGoal(),
    transitions={'no':'WAITFORREACHGOAL',
'reachA':'PICK',
'reachB':'PLACE'})

smach.StateMachine.add('PICK', Pick(),
       transitions={'pick':'PUBBGOAL'})

smach.StateMachine.add('PUBBGOAL', PubBGoal(),
       transitions={'pubB':'WAITFORREACHGOAL'})

smach.StateMachine.add('PLACE', Place(),
       transitions={'place':'WAITFORORDER'})
```

以上是整个系统的状态转换逻辑，可以根据需要添加相应的状态。以第一个状态为例，在这里定义了一个“WAITFORORDER”的状态，这个状态的类为WaitFororder，transitions代表状态的跳转。如果WaitFororder输出的结果为receive则跳转到“PUBAGOAL”，如果状态输出结果为wait，则跳转到“WAITFORORDER”状态。

The above is the overall system state transition logic, and additional states can be added as needed. Taking the first state as an example, a "WAITFORORDER" state is defined here, with the class name WaitFororder, and transitions represent the state transition. If the output result of WaitFororder is "receive", it jumps to "PUBAGOAL"; if the state output result is "wait", it jumps to the "WAITFORORDER" state.

4.2.6 启动建图

（1）启动雷达

3.2.6 Start mapping

(1) Start the LiDAR

```
roslaunch agilex_scout open_lidar.launch
```

（2）启动gmapping建图

(2) Start gmapping mapping

```
roslaunch agilex_scout gmapping.launch
```

用遥控器遥控机器人在场景中建图，执行以下指令保存地图

（1）进入~/catkin_workspace/src/agilex_scout/maps目录

```
roscd agilex_scout/maps
```

（2）运行保存地图的命令

Use the remote control to map the robot in the scene and execute the following command to save the map:

(1) Enter the ~/catkin_workspace/src/agilex_scout/maps directory

(2) Run the command to save the map

```
rosrun map_server map_saver -f map
```

> 最后面的map是生成地图的名字，在导航的时候默认是加载以map命名地图文件，如果在命名的时候取了别的名字需要修改加载地图的名字。把~catkin_workspace/src/agilex_scout/launch/navigation_4wd.launch文件中下面地图名字改为你命名的名字。
>
> The "map" at the end is the name of the generated map. By default, when navigating, it loads the map file named "map". If another name is used during naming, the name of the loaded map needs to be modified. Change the map name below

~catkin_workspace/src/agilex_scout/launch/navigation_4wd.launch to the name you want to named.

```
<node name="map_server" pkg="map_server" type="map_server" args="$(find agilex_scout)/maps/map.yaml" output="screen"
```

4.2.7 启动导航

（1）启动激光雷达

3.2.7 Start navigation

(1) Start the LiDAR

```
roslaunch agilex_scout open_lidar.launch
```

（2）启动move_base导航

启动导航之前，需要使能can模块

(2) Start move_base navigation

Before starting navigation, enable the CAN module.

```
rosrun scout_bringup setup_can2usb.bash
```

（3）启动move_base

(3) Start move_base

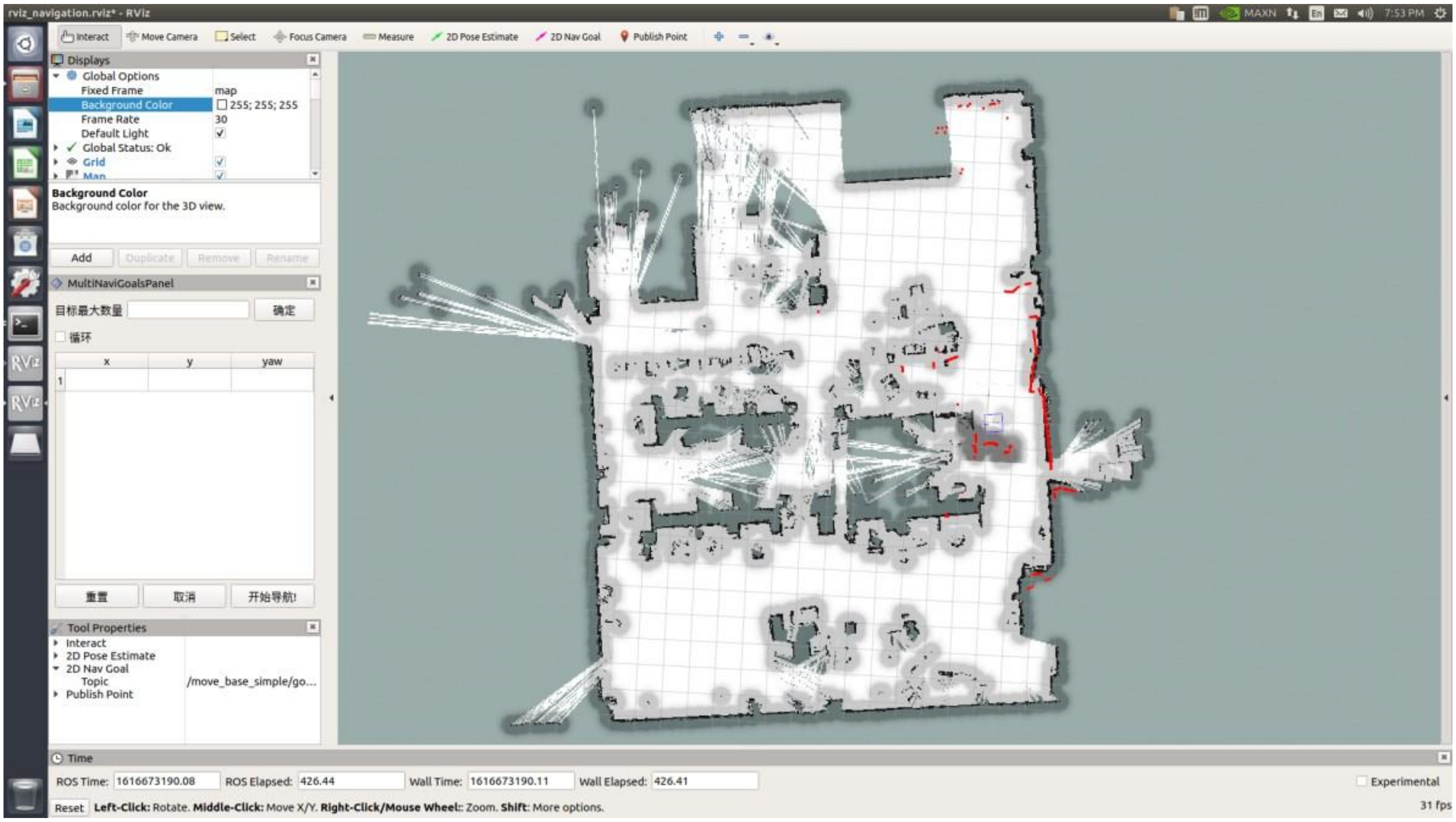
```
roslaunch agilex_scout navigation.launch
```

（4）在rviz中显示的地图上矫正底盘在场景中实际的位置，通过发布一个大概的位置加上手柄是底盘旋转校正。当激光形状和地图中的场景形状重叠的时候，校正完成

(4) Correct the actual position of the chassis in the scene on the map displayed in rviz, and calibrate the chassis rotation with a rough position and the handle. When the laser shape overlaps with the scene shape in the map, the calibration is completed.

设置定位后，激光形状和地图中的场景形状已基本重合，校正完成。如下图所示

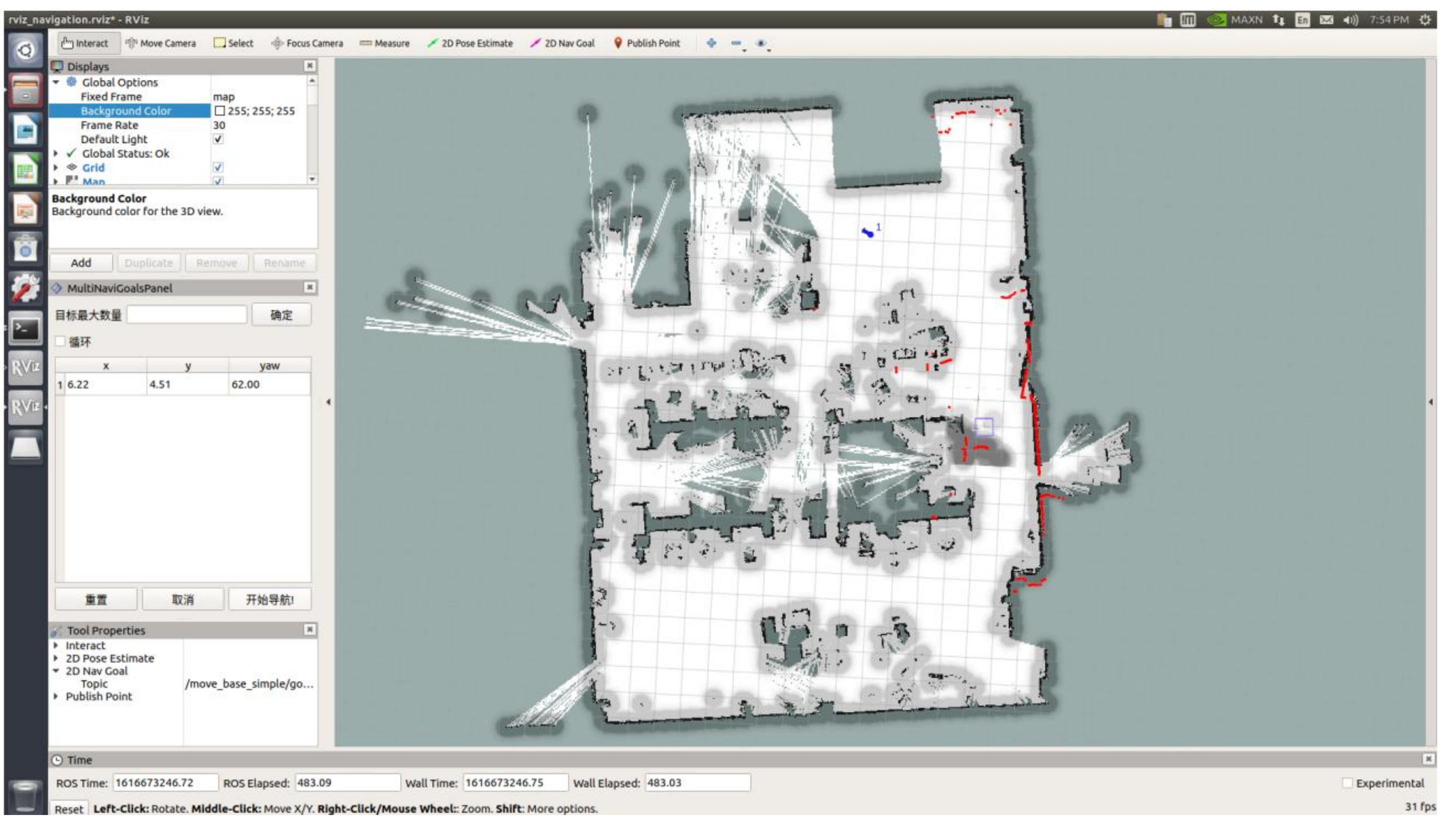
After setting the location, the laser shape and the scene shape in the map are basically overlapped, and the calibration is completed. As shown in the figure below:



（5）在左侧栏多点导航的插件上设置目标点，点击开始导航，切换手柄为指令控制模式。

(5) Set the target point in the plugin of multi-point navigation on the left sidebar, click "Start Navigation", and switch the handle to the command control mode.

已生成目标点The target point has been generated.



点击开始导航，地图已生成路径（绿色），将自动导航到目标点

After clicking "Start Navigation", the map generates a path (green) and automatically navigates to the target point.

导航的某点话题接口为/move_base_simple/goal，消息类型为geometry_msgs/PoseStamped，消息格式如下：

The topic interface for navigation at a certain point is /move_base_simple/goal, with the message type geometry_msgs/PoseStamped, and the message format is as follows:

```
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
geometry_msgs/Pose pose
  geometry_msgs/Point position
  float64 x
    float64 y
    float64 z
  geometry_msgs/Quaternion orientation
    float64 x
    float64  y
    float64  z
    float64 w
```

到达某个点之后的反馈信息可以从/move_base/result话题获得，消息类型为base_msgs/MoveBaseActionResult ，消息格式如下：

Feedback information after reaching a certain point can be obtained from the /move_base/result topic, with the message type base_msgs/MoveBaseActionResult and the message format as follows:

```
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
actionlib_msgs/GoalStatus status
  uint8 PENDING=0
  uint8 ACTIVE=1
  uint8 PREEMPTED=2
  uint8 SUCCEEDED=3
  uint8 ABORTED=4
  uint8 REJECTED=5
  uint8 PREEMPTING=6
  uint8 RECALLING=7
  uint8 RECALLED=8
  uint8 LOST=9
  actionlib_msgs/GoalID goal_id
  time stamp
    string id
```

```
  uint8 status
  string text
move_base_msgs/MoveBaseResult result
```

当机器人到达目标点后可以查看status为3。

After the robot reaches the target point, the status can be viewed as 3.

4.2.8 录制两点的坐标位置

Record the coordinates of two points

```
$  roscd agx_xarm_pick/config/
$ rosrun agx_xarm_pick record
```

需要和步骤7相互配合，把小车控制移动到指定的位置，比如工件面前，大约1m的位置。根据提示，按下回车记录两个点的位置。

It needs to cooperate with step 7 to control the car move to the specified position, such as about 1m in front of the workpiece. When prompted, press Enter to record the positions of the two points.

4.2.9 移动抓取演示

在场景中设置好两个地点作为一个夹取点和摆放点，通过步骤8录制好两个地点的坐标位置之后，开启以下指令。

3.2.9 Mobile grabbing demo

Set two locations in the scene as a gripping point and placement point, and after recording the coordinates of the two locations through step 8, open the following command.

```
$ roslaunch agx_xarm_bringup setup_arm.launch          #启动机械臂
$  roslaunch  agx_xarm_bringup open_camera.launch       #启动相机
$  roslaunch  agx_xarm_bringup open_gripper.launch      #启动夹爪
$roslaunch agx_xarm_bringup agx_xarm_bring.launch       #启动移动夹取节点
$ rosrun agx_xarm_smach smach_demo.py                   #启动状态机
$ roslaunch agilexpro open_lidar.launch                 #启动雷达
$ roslaunch agilexpro navigation_4wd.launch             #启动导航
$ roslaunch agx_xarm_pick test.launch                   #开始执行任务
```

#Start the robotic arm

#Start the camera

#Start the gripper

#Start the mobile gripping node

#Start finite-state machine

#Start LiDAR

#Start navigation

#Start executing tasks

切换遥控器为指令控制模式，则复合移动机器人开始导航到第一个地点抓取东西。

Switch the remote controller to command control mode to initiate navigation of the composite mobile robot to the first location for picking up an object.

4.2.10 原地抓取和放置演示3.2.10: In-place pick-up and placement demonstration

把抓取物体贴上标签，摆放到机器人一个合适的位置。执行以下指令就可以抓取和摆放的演示

Label an object for pick-up and place it in a suitable location for the robot. The following commands can be executed to demonstrate pick-up and placement.

```
$ roslaunch agx_xarm_bringup set_setup_arm.launch
$  roslaunch  agx_xarm_bringup open_camera.launch
$  roslaunch  agx_xarm_bringup open_gripper.launch
$ rosrun agx_xarm_pick demo_pick                #抓取物体
$ rosrun agx_xarm_pick demo_place               #放置物体
```

 # Grab the object

 # place the object

5 常见问题的处理与解答 4 Handling and Answering of Frequently Asked Questions

6 其他说明

1. xArm用户手册
2. xArm GitHub
3. SCOUT用户手册
4. AgileX GitHub

5. RoboSense-LiDAR

5 Other instructions

1. xArm User Manual

2. xArm GitHub

3. SCOUT user manual

4. AgileX GitHub

5. RoboSense-LiDAR