

# 使用手册 - PROBOT Anno 机械臂

---

Revision 1.0.0

2019-04-22

适用于 PROBOT Anno 机器人

（硬件版本号 V1.0.0）

安诺机器人（深圳）有限公司

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

## 敬告

感谢您购买和使用 PROBOT Anno 产品。为了您的安全和利益，在使用产品前请您仔细阅读本产品用户使用手册及随机附带的全部资料。如果您未按照用户使用手册操作和使用产品，而导致任何的人身伤害、财产或其他损失，安诺机器人（深圳）有限公司将不承担责任。

## 版权说明

对于您将阅览的以下信息（包括但不限于文字表述及其组合、图标、图片及图表、版面设计、编排方式、数据及软件介绍、色彩搭配等），安诺机器人（深圳）有限公司特发表以下声明：

本文档系安诺机器人（深圳）有限公司（以下简称：本公司）创作并对其享有完全的、完整的版权，未经本公司书面同意，任何单位或个人均不得以任何形式进行转载、复制、编辑、修改，或以其他方式违法使用。

本文档中可能产生的著作权、硬件、软件及专有技术的所有权、或某项技术的专利申请权、专利权等全部权利皆为本公司所有。

未经本公司书面同意，其他单位或个人使用该信息资料而影响自身或第三方权益的，或第三方未同本公司联系、核实而与其他单位或个人进行交易并造成损失的，本公司不承担任何赔偿或补偿责任。

安诺机器人（深圳）有限公司

地址：广东省深圳市宝安区西乡街道南昌第一工业区 1 栋 705

销售邮箱：[sales@robotanno.com](mailto:sales@robotanno.com)

售后邮箱：[support@robotanno.com](mailto:support@robotanno.com)

官方网站：[www.robotanno.com](http://www.robotanno.com)

## 强制

- 本说明书对 PROBOT Anno 机械臂的使用进行了全面说明，请务必在认真阅读并充分理解的基础上操作机械臂。

## 注意

- 说明书中的图及照片，为代表性示例，可能与所购买产品不同。
- 说明书有时由于产品改进、规格变更及自身更便于使用等原因而进行适当的修改，修改后的说明书将更新封面下边版本号，并以修订版发行。
- 由于破损、丢失等原因需订购说明书时，请与本公司销售部联系，按封面的版本号订购。
- 客户擅自进行产品改造，不在本公司保修范围之内，本公司概不负责。

# 目录

---

目录.....	4
文档版本.....	6
1 安全注意事项.....	7
1.1 符号及其含义.....	7
1.2 危险事项.....	7
1.3 注意事项.....	8
1.4 使用环境.....	9
1.5 安全操作规程.....	10
1.6 常规保养与存储.....	10
2 系统说明.....	12
2.1 产品概述.....	12
2.2 产品介绍.....	12
2.3 功能特性.....	13
2.4 产品外观.....	14
2.5 按钮与开关.....	15
2.6 硬件参数.....	16
2.7 结构尺寸.....	17
3 培训课程.....	19
3.1 课程特色.....	19
3.2 “古月居”简介.....	20
3.3 课程定制与培训服务.....	20
4 基本操作.....	22
4.1 启动与关闭机械臂.....	22
4.2 急停.....	22
4.3 机械臂原点位置调零.....	23
5 控制器的使用.....	25
5.1 启动 ROS2GO.....	25
5.2 上位机 ROS 环境搭建.....	26
5.3 上位机功能介绍.....	32
5.4 单机仿真.....	39
5.5 真机运行.....	40

6	MoveIt!编程接口使用说明.....	43
6.1	设置运动状态.....	43
6.2	设置目标位姿.....	47
6.3	笛卡尔空间运动.....	52
7	固件升级.....	56
7.1	概述.....	56
7.2	安装 SFU 固件升级工具.....	56
7.3	使用.....	57
7.4	SFU 命令大全.....	60
7.5	常见问题.....	61
8	常见故障诊断.....	64
8.1	无法开机.....	64
8.2	机械臂无法回到原点.....	64
8.3	机械臂抓不紧物体.....	64
8.4	上位机关闭后无法连接控制器.....	65
9	附录.....	66
9.1	参考资料.....	66
9.2	联系方式.....	66

# 文档版本

---

日期	版本	作者	概要
2019-03-27	1.0	robotanno	初始版本


Table 1: 版本历史


# 1 安全注意事项


---


感谢您购买 PROBOT Anno 机械臂。为了您的安全和防止损坏机械臂，请在使用机械臂前熟读并掌握本说明书和其他附属资料，在熟知全部设备知识、安全知识及注意事项后再开始使用，并特别注意以下安全标识。

## 1.1 符号及其含义

 **危险** 误操作时有危险，可能发生死亡或重伤事故

 **注意** 误操作时有危险，可能发生中等程度伤害、轻伤事故或物件损坏

 **强制** 手册和文档中必须遵守的事项

 **禁止** 手册和文档中明确禁止的事项

即使是属于“注意”类的事项，也会因情况不同而产生严重后果，故任何一条“注意”事项都极为重要，请务必严格遵守。

## 1.2 危险事项

 **危险**

(1) 紧急情况下，马上按下急停键，若不能及时制动机械臂，则可能引发人身伤害或设备损坏事故。



急停键

(2) 解除急停后再接通伺服电源时，要解除造成急停的事故后再启动急停键，由于误操作造成的机械臂动作，可能引发人身伤害事故。



急停状态解除、

(3) 在机械臂动作范围内运动时，请遵守以下事项：

- 1) 考虑机械臂突然向自己所处方位运动时的应变方案。
- 2) 确保设置躲避场所，以防万一。

### 注意

由于误操作造成的机械臂动作，可能引发人身伤害事故。

(4) 进行以下作业时，请确认机械臂的动作范围内没人，并且操作者处于安全位置操作：

- 1) PROBOT Anno 机械臂接通电源时。
- 2) 试运行。
- 3) 示教再现时。

(5) 请不要在机械臂工作状态下移动和维修机械臂，如要移动和维修请先关闭机械臂电源，断电之后再执行此项操作。

### 注意

不慎进入机械臂动作范围内或与机械臂发生接触，都有可能引发人身伤害事故。如发现异常时，请立即按下急停键。

急停键位于 PROBOT Anno 机械臂电器控制箱正面右侧。

## 1.3 注意事项

(1) 使用 PROBOT Anno 机械臂前要检查以下事项，如有异常则应及时修理或采取其他必要措施。

- 1) 机械臂各轴是否处于零位;
- 2) 电气线缆是否已正确连接;
- 3) 电气线缆外皮有无破损;
- 4) 急停开关是否处于解除状态;
- 5) 机械臂动作有无异常、异响。

(2) PROBOT Anno 机械臂所有者、操作者必须对自己的安全负责。安诺机器人提醒用户在使用机械臂产品时必须穿戴安全保护装备，必须遵守安全条款。

(3) 请勿改造机械臂，因擅自进行产品改造而产生的事故或故障，不在本公司保修范围之内，本公司概不负责。

(4) 请勿靠近正在运转的机械臂，以防误伤、损坏机械臂。

(5) 为了防止手动调整时的操作错误或者因安全确认不充分所导致的事故，当两个人以上操作的时候，请明确指定监督责任者。

(6) 在理解 PROBOT Anno 机械臂使用说明书的“警告标志”的基础上，使用机械臂。

## 1.4 使用环境

(1) 不要将机械臂放于恶劣环境中。泥土、废屑、高温会损坏内部器件。

(2) 机械臂可以通过网线或者 USB 线连接手机、个人电脑。此时，请确保发送的位置信息在不损坏机械臂的范围内，如果出现发送的位置信息超出安全范围，请及时关闭电源开关或者按下急停开关。

(3) 使用完机械臂，应将电源线插头拔掉，并将机械臂放于干燥、常温之处。高温及恶劣环境有损机械臂内部器件。

(4) 以下场合不可使用 PROBOT Anno 机械臂：

- 1) 靠近可燃性物质的环境
- 2) 有爆炸可能的环境
- 3) 水中或其他液体中

- 4) 存在腐蚀性、易燃性气体的环境内
- 5) 温度超过 40 摄氏度的环境
- 6) 其他恶劣使用环境

## 1.5 安全操作规程

### (1) 控制机械臂运动时

- 1) 在操作机械臂前要采用较低的倍率速度调试机械臂的运动，以增加对机械臂的有效控制。
- 2) 在按下电源键之前要考虑到机械臂的运动趋势。
- 3) 要预先考虑好避让机械臂的运动轨迹，并确认该线路不受干涉。
- 4) 机械臂周围区域必须清洁、无油，水及杂质等。

### (2) 生产运行

- 1) 在开机运行前，须知道机械臂根据所编程序将要执行的全部任务。
- 2) 须知道所有会影响机械臂移动的开关、传感器和控制信号的位置和状态。
- 3) 必须知道机械臂控制设备上的紧急停止按钮的位置，准备在紧急情况下按下这些按钮。
- 4) 永远不要认为机械臂没有移动其程序就已经完成。因为这时机械臂很有可能是在等待让它继续移动的输入信号。

## 1.6 常规保养与存储

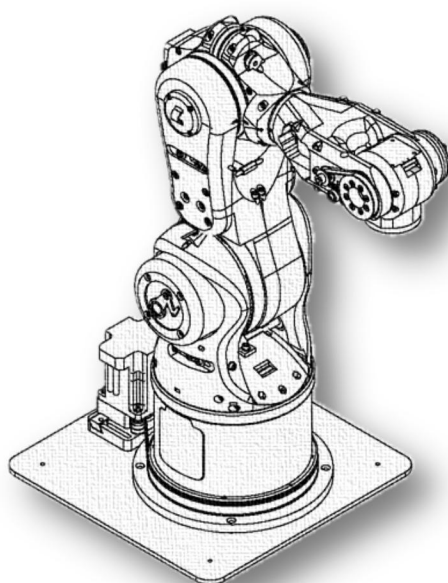
PROBOT Anno 机械臂使用安全，能最大限度的适应环境，使用时请遵照说明书的指示。请务必遵守本手册中的注意事项。

## 强制

- (1) 长期不使用，请将遥控器内部干电池取出，以免电池液泄露，腐蚀设备！
- (2) 绝不要强制地扳动机械臂的轴，否则可能会造成人身伤害和设备损坏。
- (3) 避免处在低于 20 摄氏度或高于 40 摄氏度以上温度；
- (4) 避免长期放置于阳光直射位置；
- (5) 避免处于泥土和多灰尘的环境；
- (6) 远离较强的振动环境；
- (7) 远离高湿度环境；
- (8) 远离静电环境。

### 2.1 产品概述

安诺机器人是桌面机械臂解决方案的专业研发和生产商。致力于推动传统工业机器人向智能工业机器人的转型升级，通过研发和整合芯片、操作系统、工业物联网和人工智能技术，提供新一代智能机器人解决方案。



PROBOT Anno 是安诺机器人与精锋微控推出的一款面向科研和教育领域的六轴机械臂产品，简单易用，配套教学课程，能帮助您提高研究与学习的效率。

### 2.2 产品介绍

PROBOT Anno 是一款重量轻、速度快、重复定位精度高、高性价比的消费级桌面型机械臂。采用工程塑料作为机器人本体材质，外形小巧、体积小，能够高速、高精度的完成上下料、分拣、装配等各项工作。

PROBOT Anno 提供串口、网络、IO 等丰富的通讯方式，支持 ROS、Python、C++进行二次开发，重复定位精度可达 0.5mm，搭配简单易用的可视化交互软件和仿真平台，支持多种便捷切换的末端执行器，满足不同任务需求。

## 2.3 功能特性

- 轻量化

轻量化机械结构设计，采用塑料 PLA/ABS 作为本体主要结构部件，模块化组装工艺；

- 高性能

控制器使用性能强劲的 Xilinx Zynq 芯片，ARM+FPGA 异构 SoC 让二次开发灵活高效，同时采用多轴联动插补控制算法，提高系统的精度和稳定性；

- 扩展性强

提供 ROS/Python/C++ SDK，支持 RS-232/Ethernet/Digital IO 通信扩展，包含机器视觉、语音识别等多款 AI 案例，充分发挥 PROBOT Anno 的最大潜力；

- 易用性

支持 ROS、ROS-I 框架，丰富的开源功能包，帮助用户快速上手机器人的使用和开发；

- 配套课程

标配“古月居”出品的 ROS 机械臂开发视频课程与教材，理论与实践结合，快速上手机械臂开发；同时可选课程定制、机器人线上/线下培训等增值服务，提供教育综合解决方案；

- 持续更新

支持固件更新，软件持续迭代优化，配套应用和教程不断升级。

## 2.4 产品外观

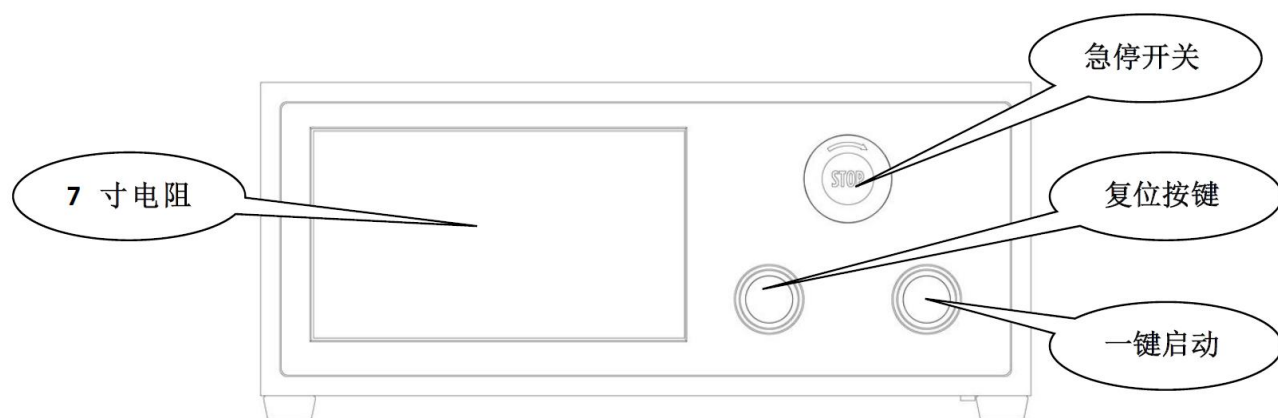


图 2-1 PROBOT Anno 机械臂外观

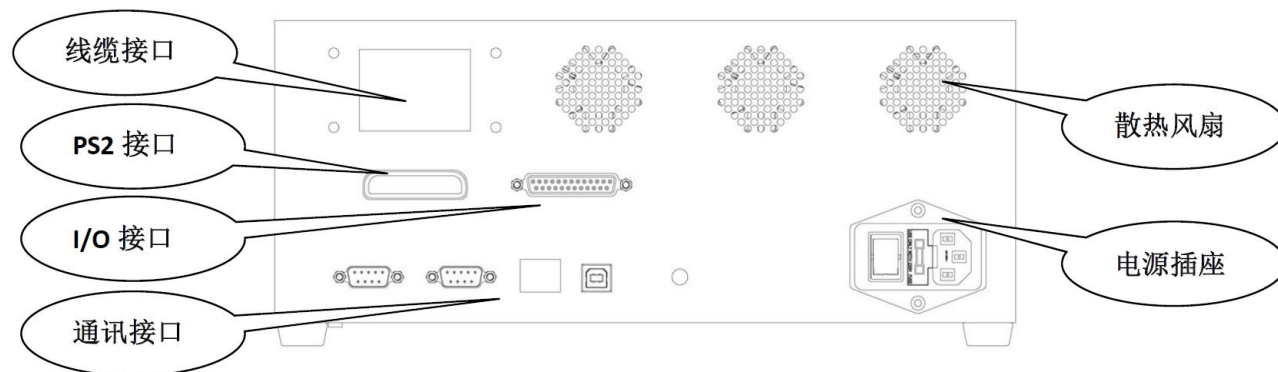


图 2-2 PROBOT Anno 机械臂电控箱外观图

## 2.5 按钮与开关



控制箱前端介绍



控制箱后端介绍

在底部控制箱正面上，从左到右分别为：急停按钮（STOP）、复位按钮（RESET）、一键启动按钮（BUTTON）；在控制箱后面有红色的电源开关。

（1）急停按钮（STOP）：在紧急情况下，向下按压急停按钮可以及时停止机械臂的运转；再次启动机械臂时需顺时针扭动该按钮，以解除急停状态。

（2）复位按钮（RESET）：直接按压有复位的作用。

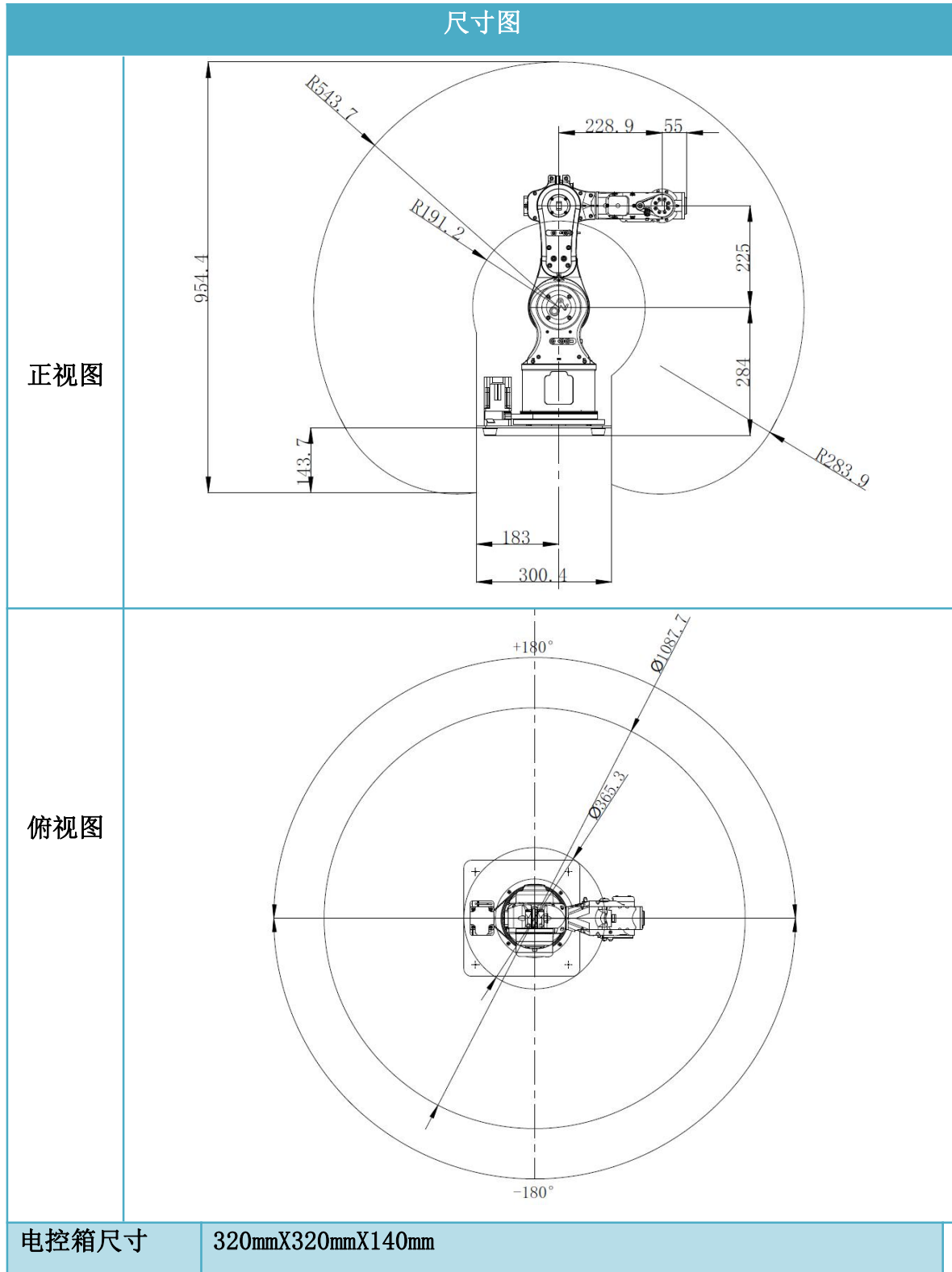
（3）一键启动按钮（BUTTON）：当按下时，执行你程序中设置的动作。

（4）电源开关：I 为开，O 为关。

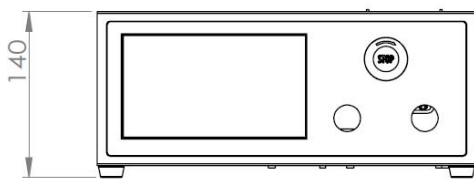
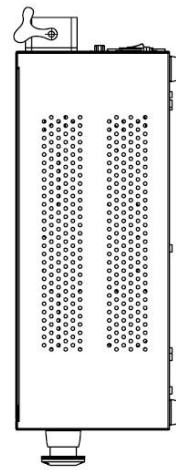
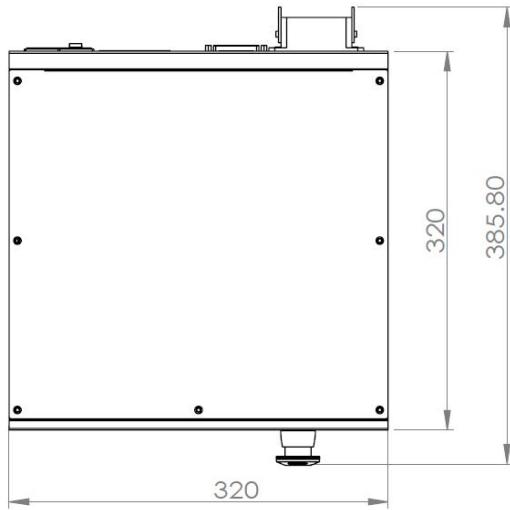
## 2.6 硬件参数

机械臂类型		六轴	
轴	运动范围	最大单轴线速度	
第 1 轴	±180°	200mm/s	
第 2 轴	±115°	200mm/s	
第 3 轴	±130°	200mm/s	
第 4 轴	±180°	200mm/s	
第 5 轴	±165°	200mm/s	
第 6 轴	±180°		
水平行程	980mm	重量 (KG)	15
垂直行程	886.3mm	结构形式	串联
电源电压	220V	减速装置	同步带
电源功率	360W	额定负载 (KG)	1
最大工作半径	482.3mm	控制器	ROBCELL A1
重复定位精度 (mm)	±0.5	本体外壳材质	工程塑料
安装方式	桌面	电机类型	步进电机

## 2.7 结构尺寸



控制箱尺寸图



## 3 培训课程

PROBOT Anno 标配“古月居”出品的 ROS 机械臂开发视频课程与教材，内容涵盖基础原理、功能实践、综合应用三篇，详细讲解了 ROS 机械臂开发中的主要流程与关键技术，理论与实践结合，快速上手机械臂开发。

### ROS机械臂开发：从入门到实战



基础原理篇	功能实践篇	综合应用篇
<b>1. ROS的过去、现在和未来</b> 1.1 ROS发展与现状 1.2 课程介绍	<b>5. 搭建仿真环境一样玩转ROS机械臂</b> 5.1 ROS中的控制器插件 5.2 构建MoveIt!+Gazebo仿真	<b>9. “手眼”结合完成物体抓取应用</b> 9.1 手眼标定 9.2 机械臂抓取
<b>2. 风靡机器人圈的ROS到底是什么</b> 2.1 通信机制 2.2 开发工具 2.3 应用功能 2.4 社区生态	<b>6. MoveIt!编程驾驭机械臂运动控制</b> 6.1 关节空间运动 6.2 笛卡尔空间运动 6.3 自主避障运动	<b>10. 针对工业应用的ROS-I又是什么</b> 10.1 ROS-I框架介绍 10.2 ROS-I应用原理 10.3 ROS-I代码浅析
<b>3. 如何从零创建一个机器人模型</b> 3.1 URDF建模 3.2 Solidworks导出模型	<b>7. MoveIt!中不得不说的“潜规则”</b> 7.1 圆弧运动规划 7.2 轨迹重定义 7.3 多轨迹连续运动 7.4 更换运动学插件	<b>11. 基于ROS设计一款机械臂控制系统</b> 11.1 ROS控制系统设计方法 11.2 PROBOT Anno控制系统案例分析
<b>4. ROS机械臂开发中的主角MoveIt!</b> 4.1 MoveIt!简介 4.2 MoveIt!可视化配置助手	<b>8. ROS机器视觉应用中的关键点</b> 8.1 ROS图像接口 8.2 摄像头内参标定 8.3 物体识别案例分析	<b>12. ROS — 机器人开发的神兵利器</b> 12.1 课程总结 12.2 进阶攻略 12.3 资源整理

图 3-1 《ROS 机械臂开发：从入门到实战》课程大纲

### 3.1 课程特色

- 全面梳理 ROS 机械臂开发中的关键知识点；
- 理论与实践结合，快速上手 ROS 机械臂开发；
- 以仿真为主，只需一台电脑即可开始学习；
- 结合实物演示，了解 ROS 机械臂实际项目开发中的主要方法；
- 深度剖析 Moveit 的使用接口，掌握 ROS 机械臂编程方法；

- 提供所有源代码和课件，提高学习效率；
- 通过作业巩固学习，微信群同步答疑解惑，夯实实践能力；
- 主讲人具备 8 年 ROS 开发经验，提供全网最优质和先进的知识服务。

## 3.2 “古月居”简介

“古月居”是领先的 ROS 机器人知识分享与教育平台（gyh.ai），自 2011 年开始持续在博客、微信公众号、知乎等互联网平台分享 ROS、机器人与人工智能等相关内容，累积访问量上千万，在线上与线下开设的培训课程多百场，参与人数上万人，2018 年出版 ROS 畅销书——《ROS 机器人开发实践》，获得 ROS 开发者的一致好评。



图 3-2 “古月居”课程案例

## 3.3 课程定制与培训服务

为满足不同用户的需求，同时可选课程定制、机器人线上/线下培训等增值服务，提供教育综合解决方案，详情欢迎来电垂询。

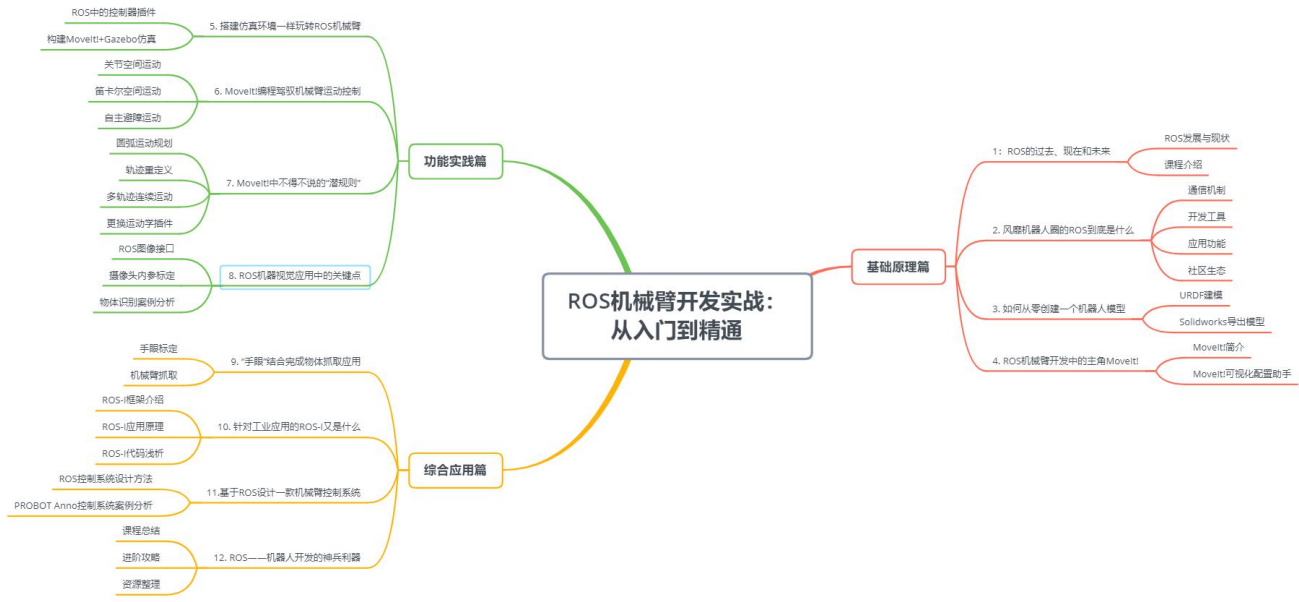


图 3-3 机械臂课程方案示例

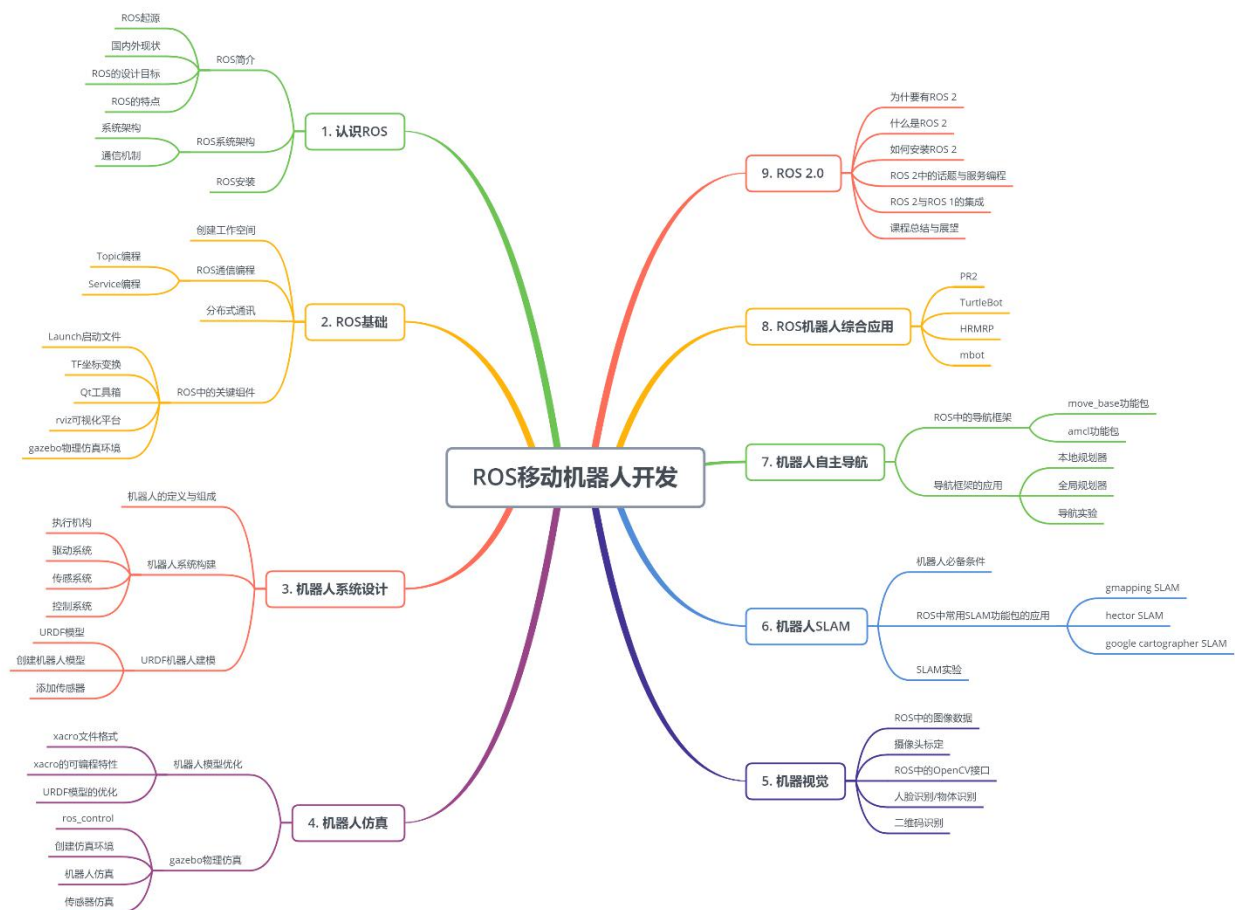


图 3-4 移动机器人课程方案示例

### 4.1 启动与关闭机械臂

PROBOT Anno 各关节的连接线已经接好，您开始使用时，只需插好电源线，打开底部控制箱的电源开关，然后解除急停按钮（当急停开关处于急停状态时），这时电源开关会发亮，表示机器能够正常启动。如果无法正常运转的话，请检查接线是否都接好、电源开关是否接触良好（接触良好时，打开电源开关会发亮）、急停按钮是否已经被按下（如果按下了请顺时针扭动以解除急停状态）。

#### 注意

- 启动机械臂前要检查机械臂外部电线遮盖物及外包装有无破损，如有有异常则应及时修理或采取其他必要措施。
- 急停按钮被按下表示处于急停状态，此时顺时针扭动可以解除急停状态。

#### 危险

- 请不要在机械臂工作状态下移动机械臂，如要移动请先关闭机械臂电源，断电之后再移动机械臂。

### 4.2 急停

在紧急情况下，向下按压急停按钮可以及时停止机械臂的运转；再次运转机械臂时顺时针扭动该按钮解除急停状态；

#### 危险

- 紧急情况下，马上按下急停键，若不能及时制动机器臂，则可能引发人身伤害或设备损坏事故。



急停键

- 请不要在机械臂工作状态下移动和维修机械臂，如要移动和维修请先关闭机械臂电源，断电之后再执行这些操作。



急停状态解除

由于急停后机械臂各轴失去力，会因为重力原因自然运动，请注意保护人身及相关设备的安全。

### 4.3 机械臂原点位置调零

#### ! 强制

在控制箱上电启动之前，必须先将机械臂回归初始位置！即将机械臂的每个轴的起始位置归零。否则机械臂极可能运行失常而导致机械臂损坏，引发危险等，初始位置如下图：



 **注意**

如果已经接通机械臂的电源并且打开电源开关了，请关掉电源开关再手动调零。

## 5 控制器的使用

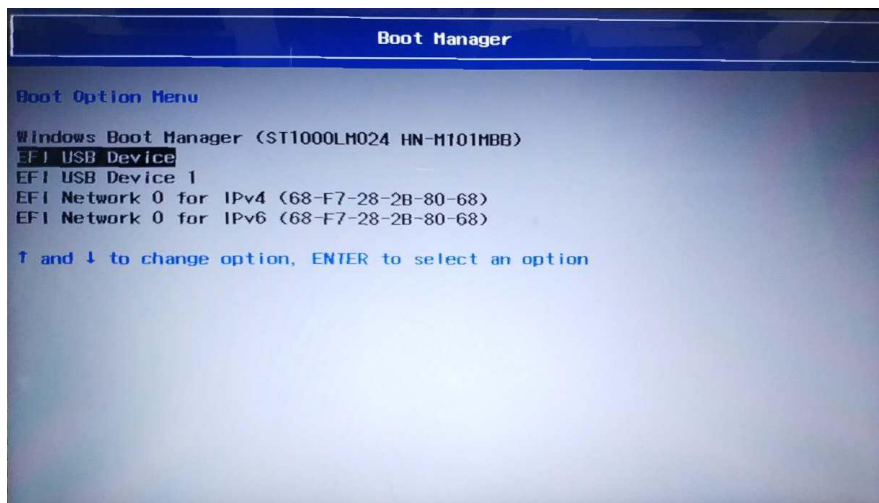
### 5.1 启动 ROS2GO

为广大用户提供了即插即用的 ROS2GO 迷你 U 盘，并在 ROS2GO 中安装部署了本产品所需的一整套 ROS 环境和上位机，用户可直接使用 ROS2GO 开始体验之旅！



ROS2GO 迷你 U 盘

(1) 将 ROS2GO 插入 PC，启动/重启 PC，并进入 BIOS 设置使用 **UEFI 模式** 引导 U 盘启动，（不同型号的 PC，进入 BIOS 系统的方式有所不同，具体请参考使用 PC 的启动方式）。



BIOS 设置 U 盘启动示意图

(2) 设置完成后，等待 ROS2GO 完成启动，启动成功后可以看到如下系统桌面。



ROS2GO 系统桌面

## 5.2 上位机 ROS 环境搭建

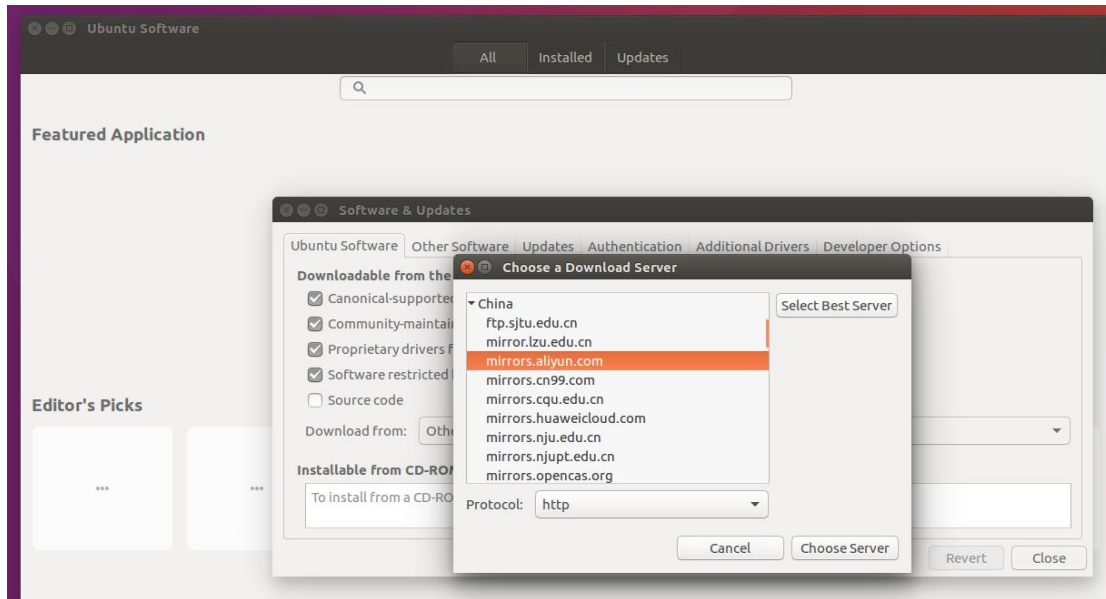
PROBOT Anno 的使用基于 ROS 环境，如果需要在自己的 PC 端安装 Ubuntu 系统和 ROS 相关的软件，请参考以下搭建步骤，另外可参考 GitHub 开源项目 README 文档：

[http://wiki.ros.org/Robots/PROBOT\\_Anno](http://wiki.ros.org/Robots/PROBOT_Anno)

### 5.2.1 安装 Ubuntu16.04 系统

(1) 可采用双系统硬盘安装或虚拟机安装，硬盘空间需在 20GB 以上，镜像下载地址为 <http://www.ubuntu.com/download/>

(2) 完成安装后建议国内用户修改系统软件源为阿里云/华为云（根据具体情况选择）：



(3) 在终端输入如下命令，更新系统软件源：

```
$ sudo apt-get update  
$ sudo apt-get upgrade
```

(4) 在终端输入如下命令，安装 Git 工具：

```
$ sudo apt-get install git
```

## 5.2.2 使用脚本安装（推荐方式）

(1) 在 Home 目录下载源码：

```
$ git clone https://github.com/ps-micro/PROBOT_Anno
```

(2) 进入/home/USER\_NAME/PROBOT\_Anno，执行安装脚本：

```
$ ./install.sh
```

运行后，将提示如下信息：

```
PROBOT Setup Assistant[v1.0.0]
---- www.ps-micro.com ----

0. Install ROS
1. Install PROBOT dependent packages
2. Install PROBOT packages
-----
3. Install all environment

[Warn] The system version: Ubuntu 16.04.5 LTS
[Warn] The ROS version: kinetic

please choose [0~3]:
```

建议输入 3，即一键安装 ROS 开发环境与上位机，或依次输入 0、1、2 分步安装对应软件。

### 注意

1. 安装完成后从 GitHub 直接获取的源代码将被复制到 `/home/USER_NAME/probot_anno_ws/src/probot_ros`，之后的开发都将基于该处源码，而 Home 目录下的 `PROBOT_Anno` 将被视为无效，因此若已确认环境安装成功可直接删除 `/home/USER_NAME/PROBOT_Anno` 文件夹；
2. 若一键安装后仍然无法正常使用，请进入 `/home/USER_NAME/PROBOT_Anno` 并再次执行安装脚本 `install.sh`。请勿运行在 `/home/USER_NAME/probot_anno_ws/src/probot_ros` 目录下的安装脚本。

### 5.2.3 使用命令安装

#### (1) 添加 ROS 软件源

`sources.list` 是 Ubuntu 系统保存软件源地址的文件，位于 `/etc/apt` 目录下，我们需要将 ROS 的软件源地址添加到该文件当中，确保后续安装可以正确找到 ROS 相关软件的下载地址。

在终端输入如下命令，添加 ROS 官方的软件源镜像：

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

#### (2) 添加密钥

在终端输入如下命令，添加密钥：

```
$ sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key
421C365BD9FF1F717815A3895523BAEEB01FA116
```

#### (3) 安装 ROS 桌面完整版

ROS 系统非常庞大，包含众多功能包、函数库和工具，所以 ROS 官方为用户提供了多种安装方式，这里我们需要使用桌面完整版安装（`Desktop-Full`）。这是最为推荐的一种安装方式，除了包含 ROS 的基础功能（核心功能包、构建工具和通信机制），还包含有丰富的机器人通用函数库、功能包（2D/3D 感知功能、机器人地图建模、自主导航等）以及工具（`rviz` 可视化工具、`gazebo` 仿真环境、`rqt` 工具箱等）。

在终端输入如下命令，安装 ROS 桌面完整版：

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



无论使用哪种安装方式，都不可能将 ROS 社区内的所有功能包安装到电脑上，在后期的使用中可根据需求独立安装所需的功能包，命令格式如下：

```
$ sudo apt-get install ros-kinetic-PACKAGE
```

其中 PACKAGE 为功能包名，例如 MoveIt!功能包安装命令如下：

```
$ sudo apt-get install ros-kinetic-moveit-full
```

#### （4）安装 ROS 依赖工具

rosdep 是 ROS 中自带的工具，主要功能是为某些功能包安装系统依赖，同时也是某些 ROS 核心功能包必须要用到的工具。完成以上安装步骤后，需使用如下命令进行初始化和更新：

```
$ sudo rosdep init
```

```
$ rosdep update
```

#### （5）配置环境变量

现在 ROS 已经成功安装到电脑中了，默认在/opt 路径下。由于在后续使用中，会频繁使用终端输入 ROS 命令，所以在使用之前，还需对环境变量进行简单的设置，让系统能够通过环境变量找到命令的所在位置。

Ubuntu 默认使用的终端是 bash，在 bash 中设置 ROS 环境变量的命令如下：

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

```
$ source ~/.bashrc
```

如果你使用的是 zsh，则需要将以上命令中的 bash 都修改为 zsh：

```
$ echo "source /opt/ros/kinetic/setup.zsh" >> ~/.zshrc
```

```
$ source ~/.zshrc
```

这种方法只对当前终端有效，若重新打开一个终端，还是会默认使用 `bash` 或 `zsh` 配置文件中设置的环境变量。一种全局有效修改的方法是打开 `~/.bashrc` 或者 `~/.zshrc` 文件，找到设置环境变量的命令，修改对应的 ROS 版本并保存退出。

## (6) 安装依赖包

此外还有一些后续功能需要依赖的功能包，需要依次安装：

```
$ ROS_VERSION=`/usr/bin/rosversion -d`
$ sudo apt-get install ros-${ROS_VERSION}-moveit-*
$ sudo apt-get install ros-${ROS_VERSION}-industrial-*
$ sudo apt-get install ros-${ROS_VERSION}-gazebo-ros-control
$ sudo apt-get install ros-${ROS_VERSION}-ros-control ros-${ROS_VERSION}-ros-controllers
$ sudo apt-get install ros-${ROS_VERSION}-trac-ik-kinematics-plugin
$ sudo apt-get install ros-${ROS_VERSION}-usb-cam
```

## (7) 下载、编译、安装上位机

在 `Home` 目录下创建文件夹 `probot_ws/src`，在该目录下下载源码：

```
$ git clone https://github.com/ps-micro/PROBOT_Anno
```

回到 `probot_ws` 的路径下，使用以下命令进行编译上位机：

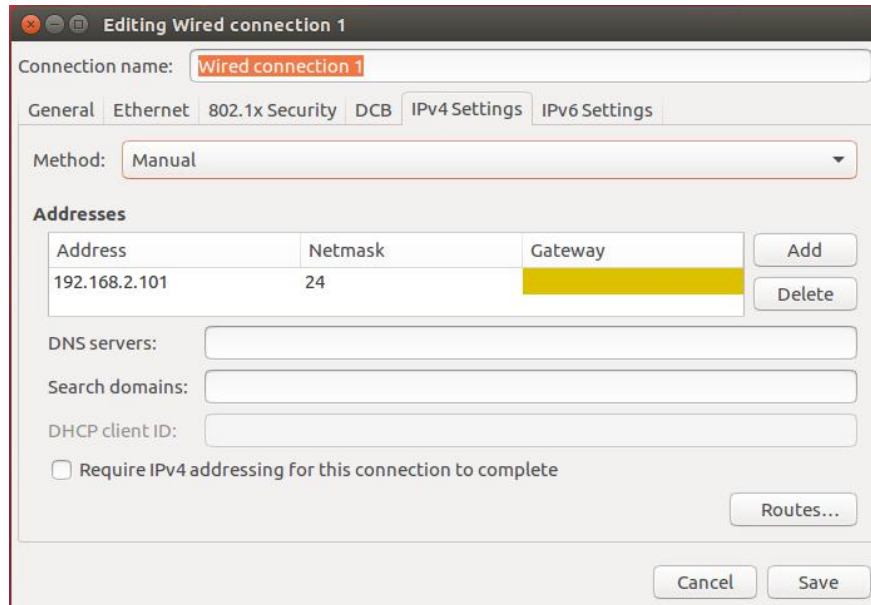
```
$/catkin_make
```

## (8) 设置工作空间的环境变量：

```
$ echo "source ~/'WORKSPACE_PATH'/install/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
```

## 5.2.4 设置 PC 端静态 IP

PC 与机械臂通过网线通信，机械臂的默认 IP 是 192.168.2.123，为保证通信连接，请在 PC 端配置静态 IP，地址网段在 192.168.2.XXX。



## 5.3 上位机功能介绍

首先，介绍上位机主界面分区及其功能，共分为 6 个区域：

- 可视化插件配置区：
- ROS MoveIt!可视化控制区：
- 控制按键区：

仿真/真机环境切换、机器人控制使能/除能、停止键、状态指示灯；

- 模型动态显示区：

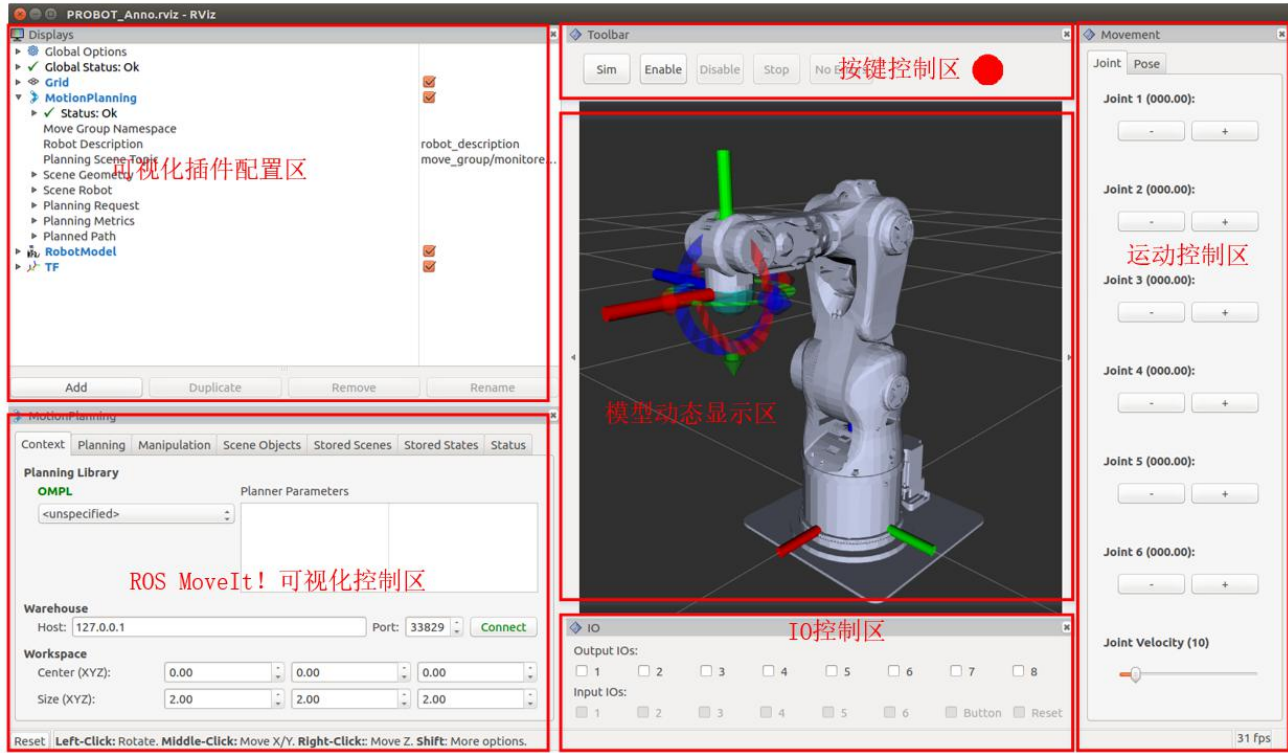
显示机器人当前实际位姿以及目标位姿，使用鼠标拖拽机器人末端拖动球可改变目标位姿；

- IO 控制区：

数字 IO 的输入输出控制；

- 运动控制区：

关节空间点动和工作空间点动。

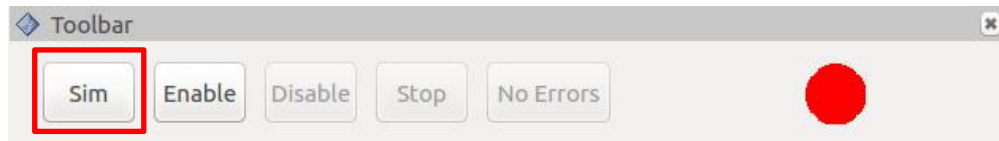


### 5.3.1 控制栏按键功能

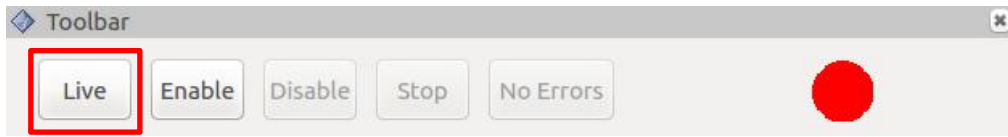
#### (1) 仿真/真机环境切换

点击该按钮可进行仿真/真机环境的切换，请务必在 Disable 状态下执行切换操作。

Sim 表示当前处于仿真环境：



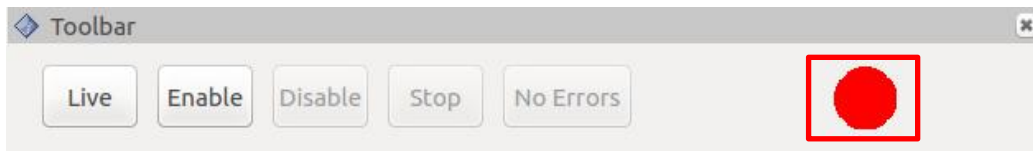
Live 表示当前处于真机控制环境：



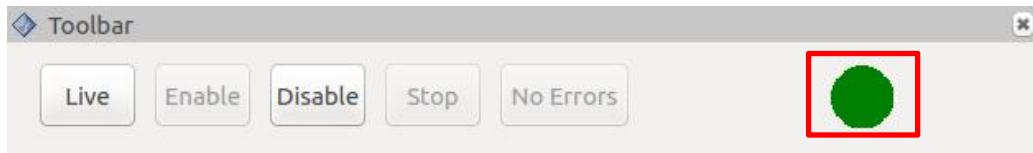
## (2) 机器人控制使能/除能

点击 **Enable** 可使能机器人控制，点击 **Disable** 除能（在仿真环境下无需使能）。

红灯表示当前机器人处于控制除能状态：



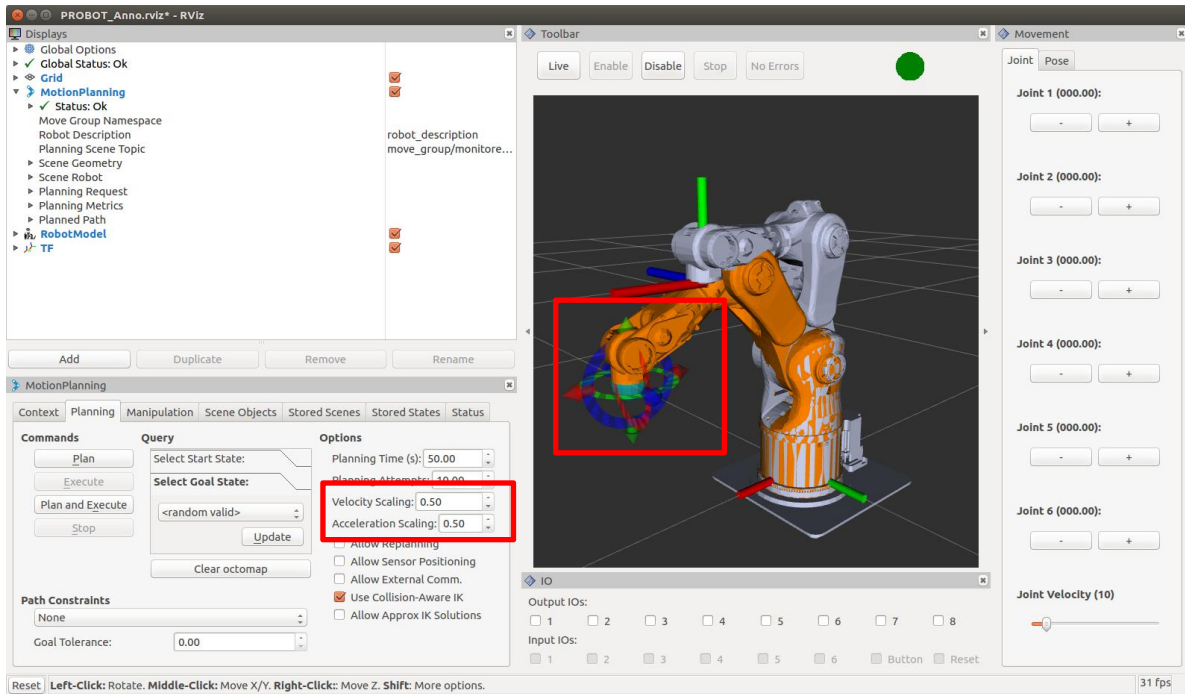
绿灯表示当前机器人处于控制使能状态：



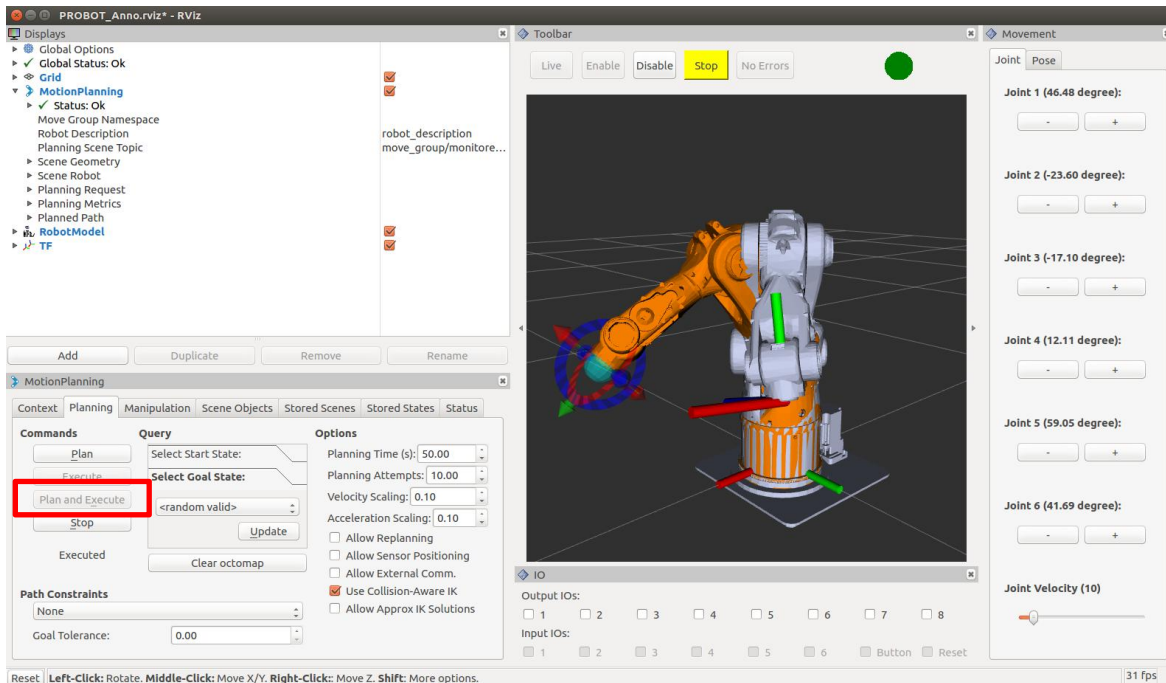
### 5.3.2 拖动施教

(1) 将鼠标定位在机器人模型终端的控制球上，长按鼠标左键可拖拽机器人，鼠标松开时显示黄色的机器人模型即为目标位姿，而银白色的机器人模型为当前实际位姿。

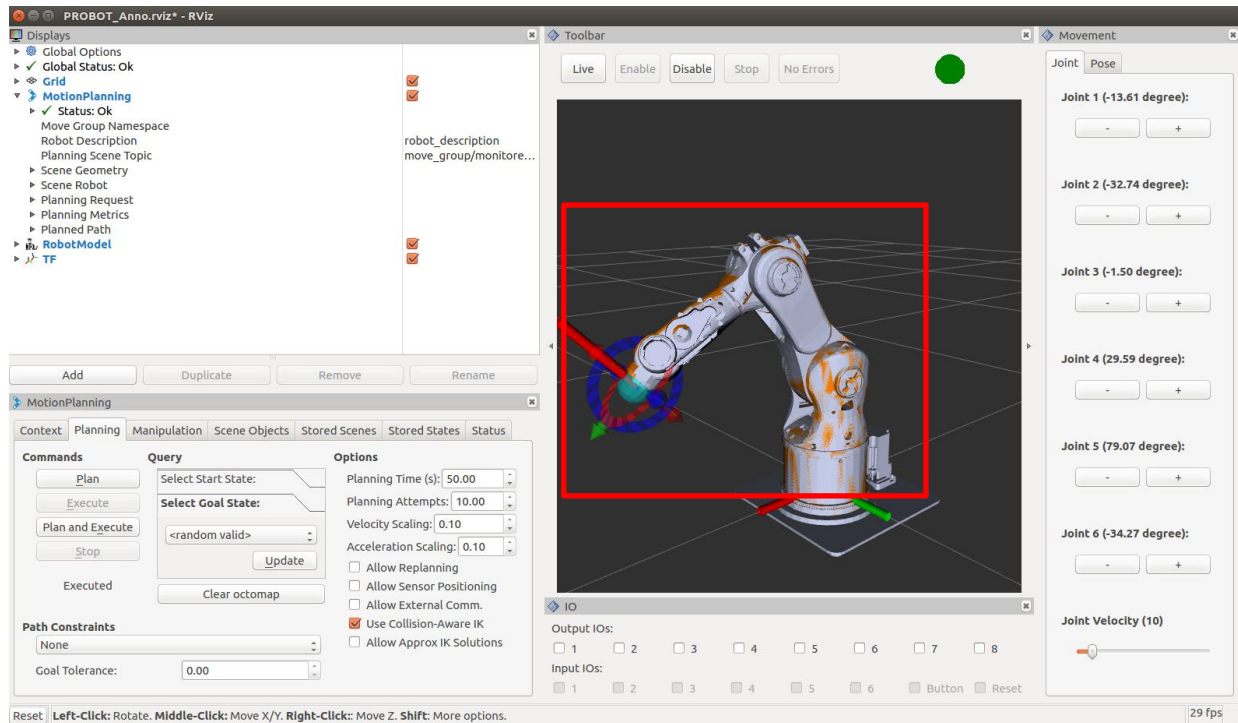
(2) 设置规划的最大时间限制和机器人运动的速度和加速度（百分比，1 表示 100%，即设置为最大值）。



(3) 点击 ROS MoveIt!可视化控制区的 Planning 标签页中的 Plan and Execute 按键，机器人模型即开始运动（运动过程中，Plan and Execute 按键变灰并且不可操作，需等待当前动作完成）。

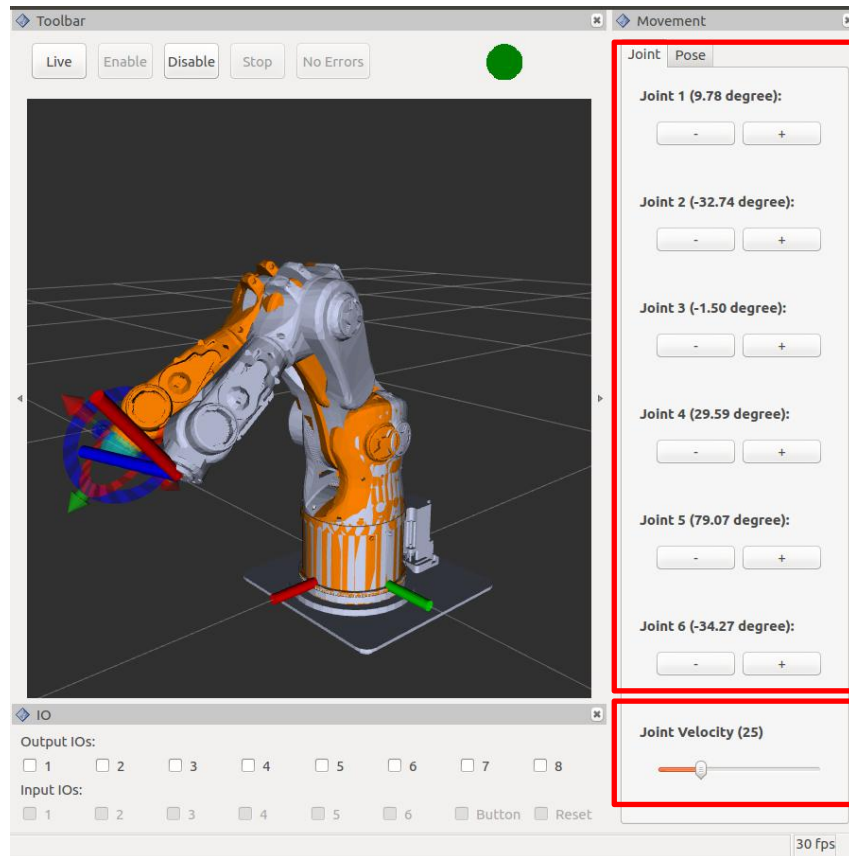


(4) 机器人从当前位姿运动到目标位姿，直到界面显示两个模型完全重合，机器人运动结束。



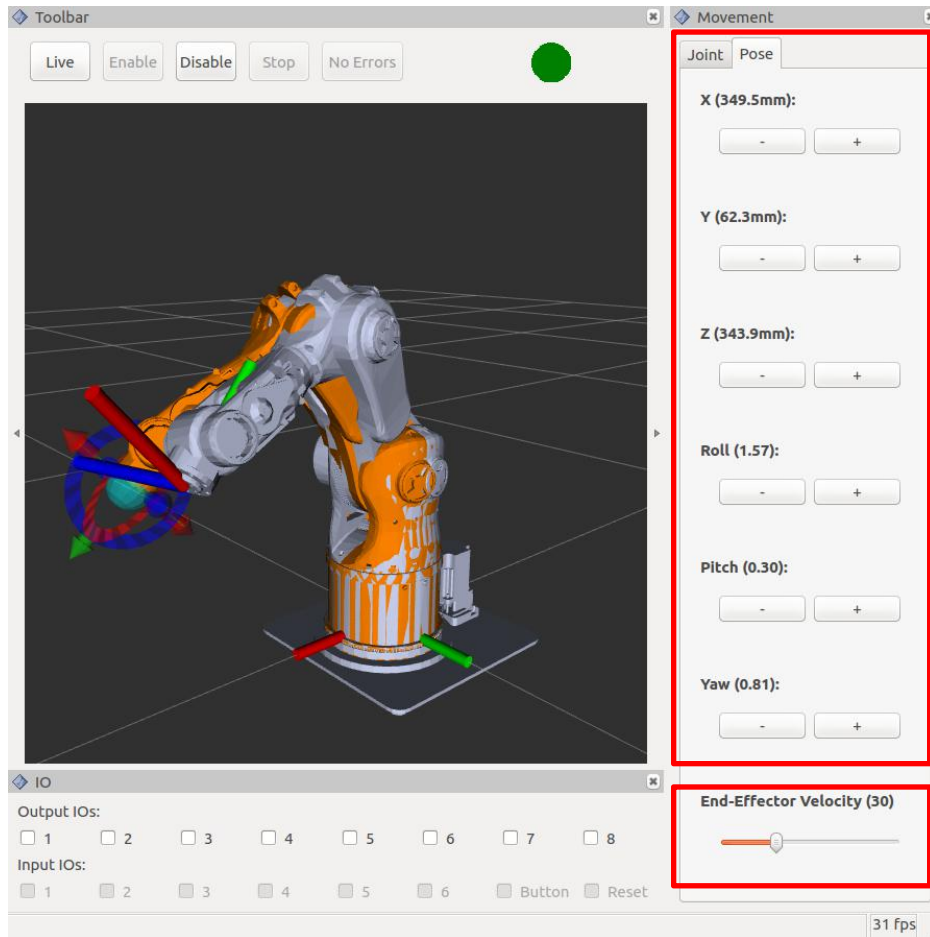
### 5.3.3 关节空间点动

- (1) 关节点动的所有功能在右侧运动控制区的 Joint 标签页中。
- (2) 通过长按+、-可控制关节运动，下侧的 Joint Velocity 滑条可调节运动速度。



### 5.3.4 工作空间点动

- (1) 空间点动的所有功能在右侧运动控制区的 Pose 标签页中。
- (2) 通过长按+、-可控制机器人末端执行工作空间点动，下侧的 End-Effector Velocity 滑条可调节运动速度。

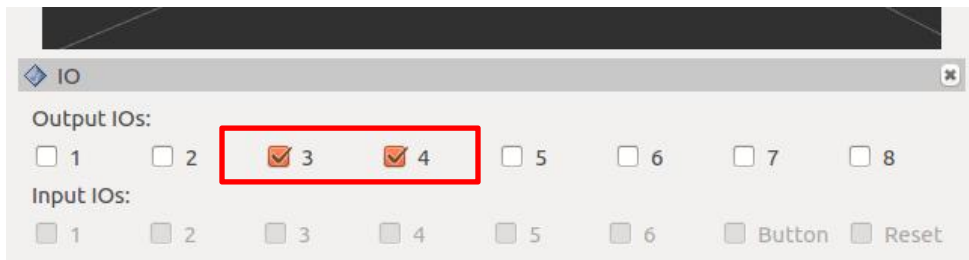


### 5.3.5 IO 控制

IO 控制功能在 IO 控制区。

(1) 鼠标左击想要控制的输出端口 (Output IOs)，将改变其输出 (0/1)，端口号前的方框中打钩代表使能低电平输出；

(2) 在 Input IOs 一栏，可看到输入端口的电平信号 (0/1)，端口号前的方框中打钩代表输入为低电平。

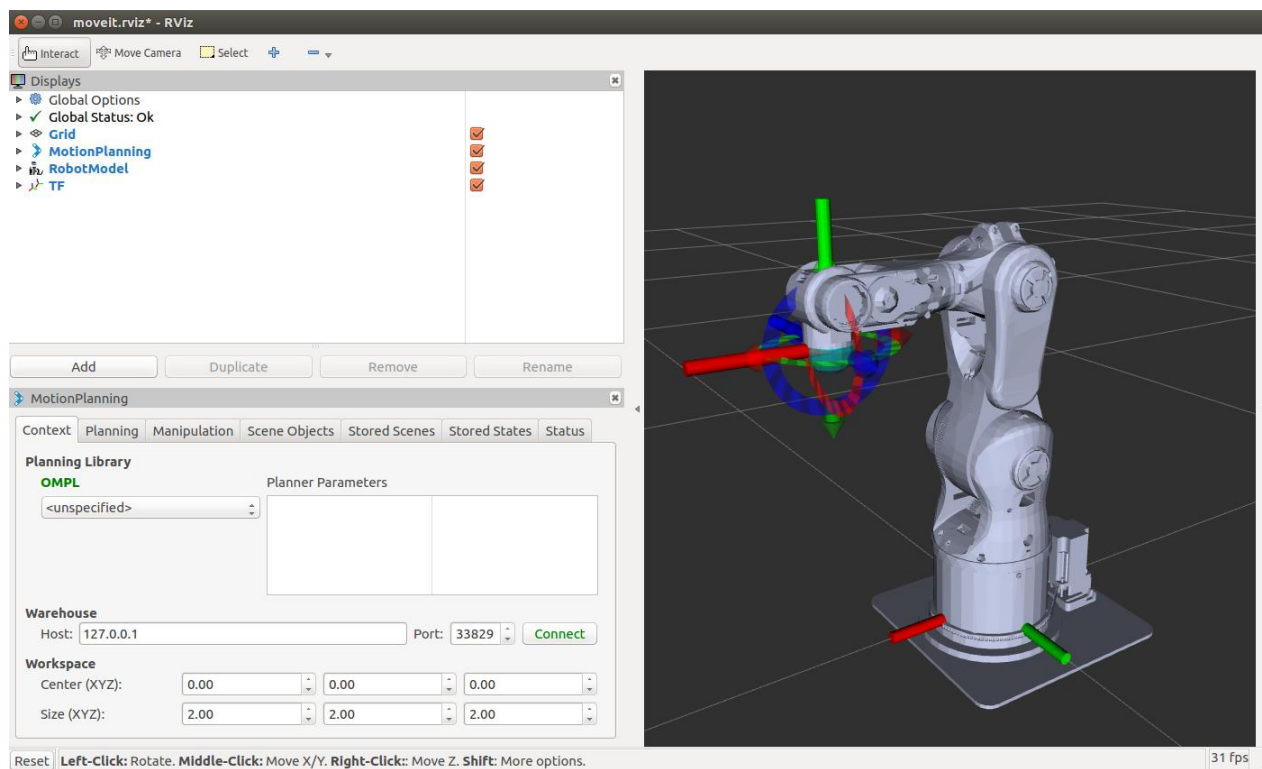


## 5.4 单机仿真

上位机 ROS 环境搭建完成后即可在 PC 上进行单机仿真，无需连接控制器和实际机器人。

使用以下命令，启动上位机并进入仿真控制环境：

```
$ roslaunch probot_bringup probot_anno_bringup.launch sim:=true
```



### 注意

使用以上命令启动的上位机界面和真机运行下启动的界面有所差别，更加精简。

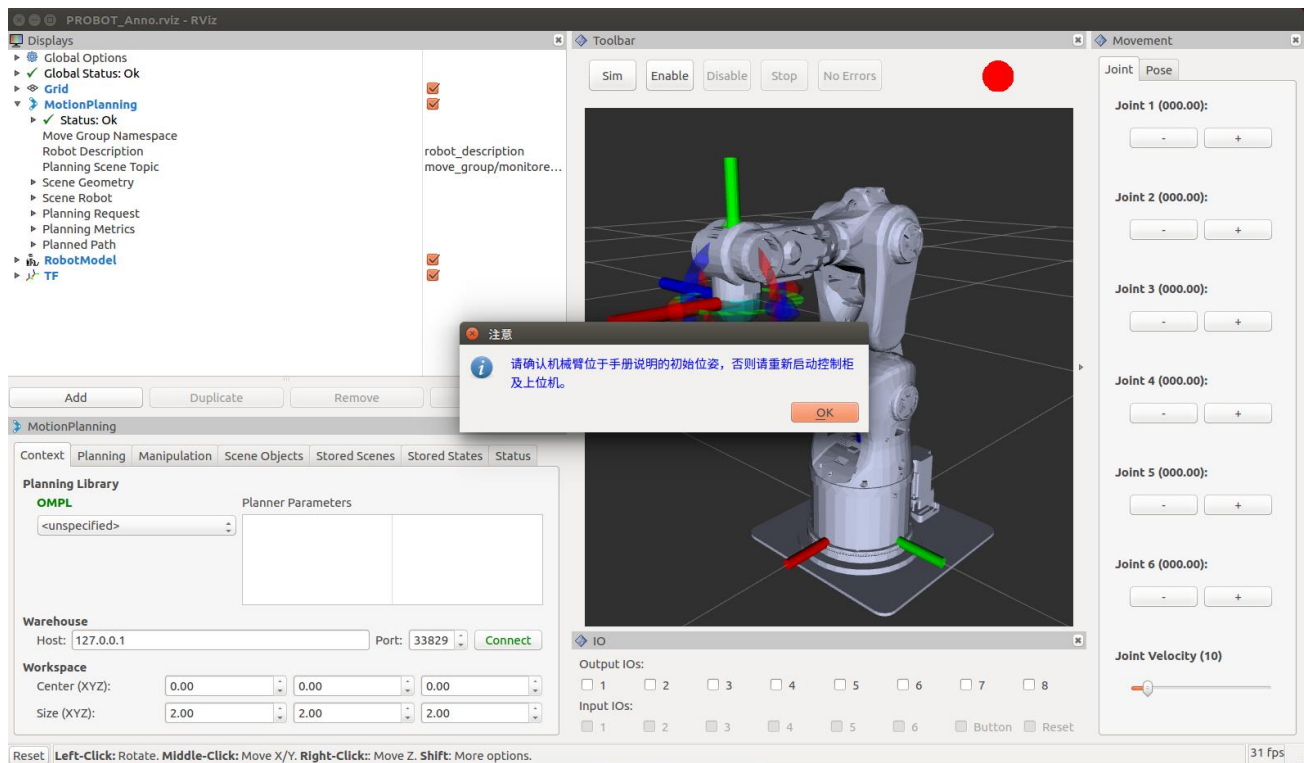
## 5.5 真机运行

### 注意

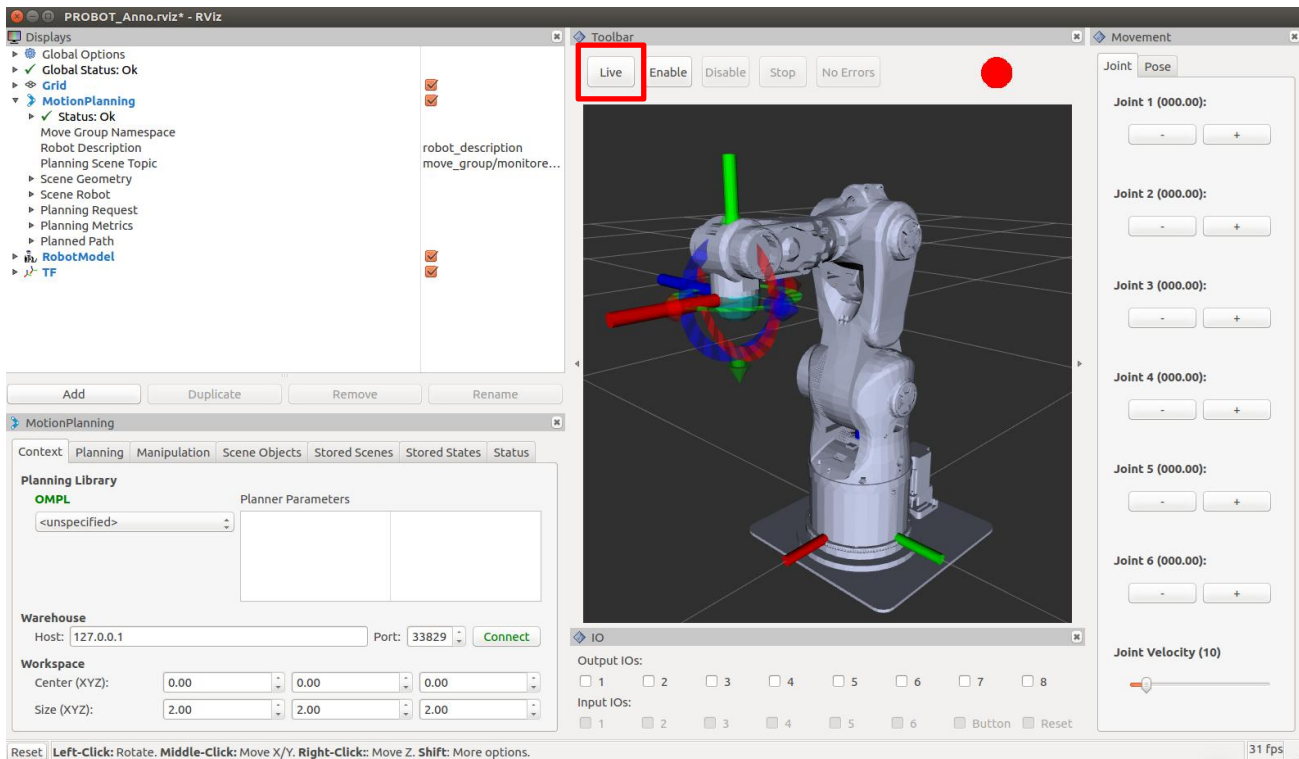
控制器上电后，PROBOT 软硬件系统需要约 20s 启动时间，启动完成后蜂鸣器将连续响两声。请确保听到连续响声后再启动 ROS 上位机！否则需要将控制器重新上电，并重启 ROS 上位机。

(1) 系统启动完成后，使用如下命令启动 ROS 上位机，请确认机械臂位于初始位姿：

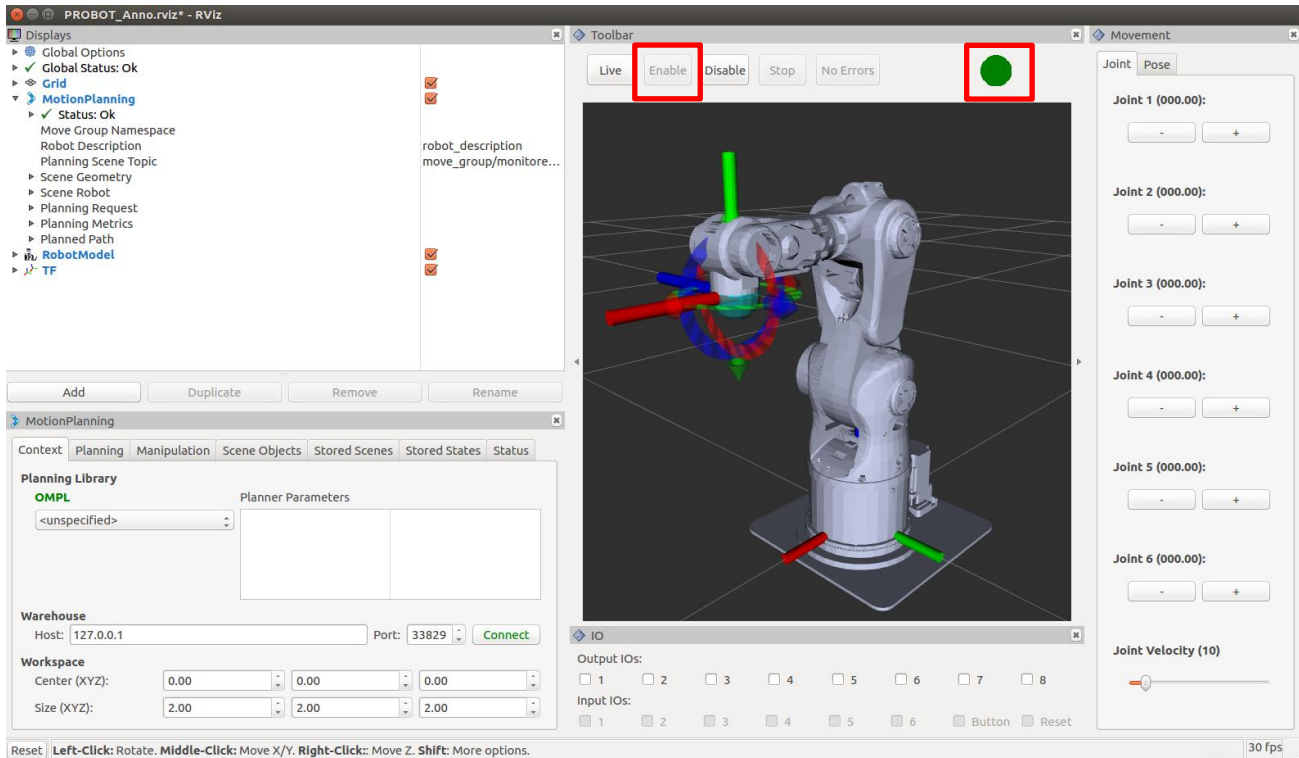
```
$ roslaunch probot_bringup probot_anno_bringup.launch robot_ip:=192.168.2.123
```



(2) 上位机启动后默认进入仿真运行，需点击控制栏中的仿真/真机切换按钮，切换到真机控制环境，按键上显示 Live:



(3) 点击控制栏的使能/除能按键，使能机器人控制，红色提示灯变为绿色：



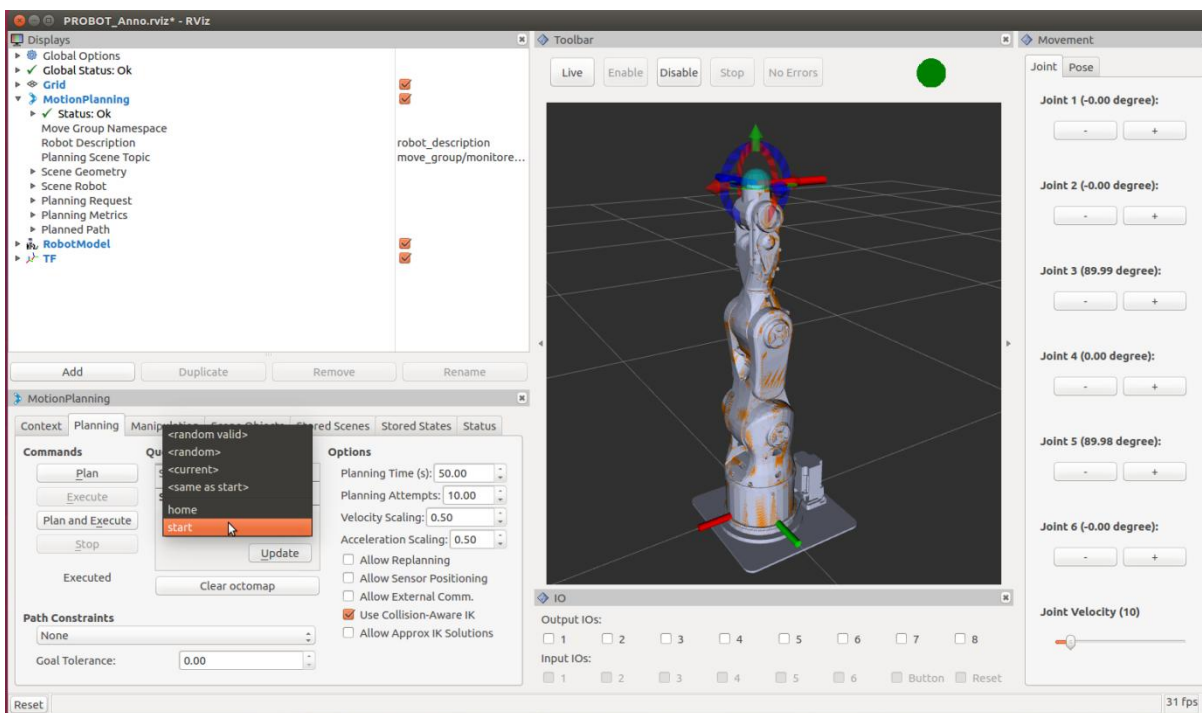
(4) 可根据 5.3 小节在界面上进行机器人控制操作！

(5) 打开一个终端，输入以下命令运行 Demo 脚本，控制机器人完成相应动作：

```
$ rosrn probot_demo test_move.py
```

## 5.6 关机与急停

操作结束后，请按照下图说明，控制机械臂回到初始位姿，然后点击“Disable”，按下控制柜急停按键后，关闭控制柜电源，完成关机。



### ⚠ 注意

运动过程中如果出现意外运动，请及时按下急停键，由于急停后机械臂各轴失去力，会因为重力原因自然运动，请注意保护人身及相关设备的安全。

# 6 MoveIt!编程接口使用说明

PROBOT Anno 兼容 ROS MoveIt!编程接口，可以在 ROS 环境下使用 MoveIt!实现机器人的应用功能开发。

## 注意

(1) 以下函数接口均具备多态特性，详细函数说明请见 MoveIt!官方介绍：

[https://docs.ros.org/api/moveit\\_ros\\_planning\\_interface/html/namespacemoveit\\_1\\_1planning\\_interface.html](https://docs.ros.org/api/moveit_ros_planning_interface/html/namespacemoveit_1_1planning_interface.html)

(2) 完整例程请参见：

[https://github.com/ps-micro/PROBOT\\_Anno/tree/master/probot\\_demo/src](https://github.com/ps-micro/PROBOT_Anno/tree/master/probot_demo/src)

## 6.1 设置运动状态

(1) void

`moveit::planning_interface::MoveGroupInterface::MoveGroupInterfaceImpl::setMaxAccelerationScalingFactor ( double max_acceleration_scaling_factor )`

函数功能	设置运动加速度系数		
函数参数	类型	名称	描述
	double	max_acceleration_scaling_factor	加速度上限系数，范围：0~1
返回值	值	描述	
	-	-	

说明	
----	--

例程:

```
moveit::planning_interface::MoveGroupInterface arm("manipulator");
arm.setMaxAccelerationScalingFactor(0.2);
```

(2) void

moveit::planning\_interface::MoveGroupInterface::MoveGroupInterfaceImpl::setMaxVelocityScalingFactor ( double max\_velocity\_scaling\_factor )

函数功能	设置运动速度系数		
函数参数	类型	名称	描述
	double	max_velocity_scaling_factor	速度上限系数，范围：0~1
返回值	值	描述	
	-	-	
说明			

例程:

```
moveit::planning_interface::MoveGroupInterface arm("manipulator");
arm.setMaxVelocityScalingFactor(0.2);
```

(3) void

moveit::planning\_interface::MoveGroupInterface::MoveGroupInterfaceImpl::setGoalJointTolerance (double tolerance)

函数功能	设置关节运动的允许误差		
函数参数	类型	名称	描述
	double	tolerance	允许误差，单位：Rad
返回值	值	描述	
	-	-	
说明			

例程：

```
moveit::planning_interface::MoveGroupInterface arm("manipulator");  
arm.setGoalJointTolerance(0.001);
```

(4) void

moveit::planning\_interface::MoveGroupInterface::MoveGroupInterfaceImpl::setGoalPositionTolerance (double tolerance)

函数功能	设置运动姿态的允许误差		
函数参数	类型	名称	描述
	double	tolerance	允许误差

返回值	值	描述
	-	-
说明		

例程：

```
moveit::planning_interface::MoveGroupInterface arm("manipulator");
arm.setGoalPositionTolerance (0.001);
```

(5) void

moveit::planning\_interface::MoveGroupInterface::MoveGroupInterfaceImpl::setGoalPositionTolerance ( double tolerance )

函数功能	设置运动位置的允许误差		
函数参数	类型	名称	描述
	double	tolerance	允许误差
返回值	值	描述	
	-	-	
说明			

例程：

```
moveit::planning_interface::MoveGroupInterface arm("manipulator");
```

```
arm.setGoalPositionTolerance (0.001);
```

(6) void moveit::planning\_interface::MoveGroupInterface::stop ( void )

函数功能	停止机器人的所有运动		
函数参数	类型	名称	描述
	-	-	-
返回值	值	描述	
	-	-	
说明			

## 6.2 设置目标位姿

( 1 ) bool moveit::planning\_interface::MoveGroupInterface::setJointValueTarget (const std::vector< double > & group\_variable\_values)

函数功能	设置各关节目标位姿		
函数参数	类型	名称	描述
	std::vector< double > &	group_variable_values	各关节位置值，单位：Rad
返回值	值	描述	

	True	设置成功
	False	设置失败
说明		

例程:

```

moveit::planning_interface::MoveGroupInterface arm("manipulator");

double targetPose[6] = {0.391410, -0.676384, -0.376217, 0.0, 1.052834, 0.454125};
std::vector<double> joint_group_positions(6);
joint_group_positions[0] = targetPose[0];
joint_group_positions[1] = targetPose[1];
joint_group_positions[2] = targetPose[2];
joint_group_positions[3] = targetPose[3];
joint_group_positions[4] = targetPose[4];
joint_group_positions[5] = targetPose[5];

arm.setJointValueTarget(joint_group_positions);
arm.move();

```

(2) `bool moveit::planning_interface::MoveGroupInterface::setPoseTarget (const geometry_msgs::Pose & end_effector_pose, const std::string & end_effector_link = "")`

函数功能	设置机器人终端的目标位姿		
函数参数	类型	名称	描述

	geometry_msgs::Pose &	end_effector_pose	终端的目标姿态
	std::string &	end_effector_link	终端名称，可缺省
返回值	值	描述	
	True	设置成功	
	False	设置失败	
说明			

例程：

```

moveit::planning_interface::MoveGroupInterface arm("manipulator");

// 设置机器人终端的目标位置
geometry_msgs::Pose target_pose;
target_pose.orientation.x = 0.70692;
target_pose.orientation.y = 0.0;
target_pose.orientation.z = 0.0;
target_pose.orientation.w = 0.70729;

target_pose.position.x = 0.2593;
target_pose.position.y = 0.0636;
target_pose.position.z = 0.1787;

arm.setPoseTarget(target_pose);
arm.move();

```

(3) void moveit::planning\_interface::MoveGroupInterface::setRandomTarget()

函数功能	设置随机目标位姿		
函数参数	类型	名称	描述
	-	-	-
返回值	值	描述	
	-	-	
说明			

例程：

```
moveit::planning_interface::MoveGroupInterface arm("manipulator");  
  
// 随机产生一个目标位置  
arm.setRandomTarget();  
  
// 开始运动规划，并且让机械臂移动到目标位置  
arm.move();
```

(4) bool moveit::planning\_interface::MoveGroupInterface::setPositionTarget ( double x, double y, double z, const std::string & end\_effector\_link = "" )

函数功能	设置机器人终端的目标位置
------	--------------

	类型	名称	描述
函数参数	double	x	X 坐标值
	double	y	Y 坐标值
	double	z	Z 坐标值
	std::string &	end_effector_link	终端名称, 可缺省
返回值	值	描述	
	True	设置成功	
	False	设置失败	
说明			

( 5 ) `bool moveit::planning_interface::MoveGroupInterface::setRPYTarget ( double roll, double pitch, double yaw, const std::string & end_effector_link = "" )`

函数功能	设置机器人终端的目标姿态		
	类型	名称	描述
函数参数	double	roll	X 轴旋转值
	double	pitch	Y 轴旋转值
	double	yaw	Z 轴旋转值
	std::string &	end_effector_link	终端名称, 可缺省

	值	描述
返回值	True	设置成功
	False	设置失败
说明		

### 6.3 笛卡尔空间运动

```
(1) double moveit::planning_interface::MoveGroupInterface::computeCartesianPath (
const std::vector< geometry_msgs::Pose > & waypoints,
double eef_step,
double jump_threshold,
moveit_msgs::RobotTrajectory & trajectory,
bool avoid_collisions = true,
moveit_msgs::MoveItErrorCodes * error_code = NULL )
```

函数功能	设置机器人笛卡尔路径		
函数参数	类型	名称	描述
	std::vector< geometry_msgs::Pose > &	waypoints	笛卡尔路径点
	double	eef_step	终端的步进值
	double	jump_threshold	跳跃阈值
	moveit_msgs::RobotTrajectory &	trajectory	规划得到的笛卡尔路径

	bool	avoid_collisions	是否允许碰撞检测
	moveit_msgs::MoveItErrorCodes *	error_code	返回的错误码
返回值	值	描述	
	double	可规划路径的覆盖率，范围：0 ~ 1	
说明			

示例：

```

moveit::planning_interface::MoveGroupInterface arm("manipulator");

//获取终端 link 的名称
std::string end_effector_link = arm.getEndEffectorLink();

// 获取当前位姿数据最为机械臂运动的起始位姿
geometry_msgs::Pose start_pose = arm.getCurrentPose(end_effector_link).pose;

std::vector<geometry_msgs::Pose> waypoints;

//将初始位姿加入路点列表
waypoints.push_back(start_pose);

start_pose.position.z -= 0.2;
waypoints.push_back(start_pose);

start_pose.position.x += 0.1;
waypoints.push_back(start_pose);

```

```

start_pose.position.y += 0.1;
waypoints.push_back(start_pose);

// 笛卡尔空间下的路径规划
moveit_msgs::RobotTrajectory trajectory;
const double jump_threshold = 0.0;
const double eef_step = 0.01;
double fraction = 0.0;
int maxtries = 100; //最大尝试规划次数
int attempts = 0;   //已经尝试规划次数

while(fraction < 1.0 && attempts < maxtries)
{
    fraction = arm.computeCartesianPath(waypoints, eef_step, jump_threshold, trajectory);
    attempts++;

    if(attempts % 10 == 0)
        ROS_INFO("Still trying after %d attempts...", attempts);
}

if(fraction == 1)
{
    ROS_INFO("Path computed successfully. Moving the arm.");

    // 生成机械臂的运动规划数据
    moveit::planning_interface::MoveGroupInterface::Plan plan;
    plan.trajectory_ = trajectory;

    // 执行运动

```

```
        arm.execute(plan);
    sleep(1);
}
else
{
    ROS_INFO("Path planning failed with only %0.6f success after %d attempts.", fraction,
maxtries);
}
```

# 7 固件升级

## 7.1 概述

PROBOT SFU 固件升级工具用于在网络环境下，为 ROBCELL 控制器提供远程固件更新服务，用户可通过该工具更新控制器设备固件。

PROBOT SFU 固件升级系统包含了 PC 端 SFU Server、SFU Commander 和控制器的 SFU Client 组件，其系统组成如下：

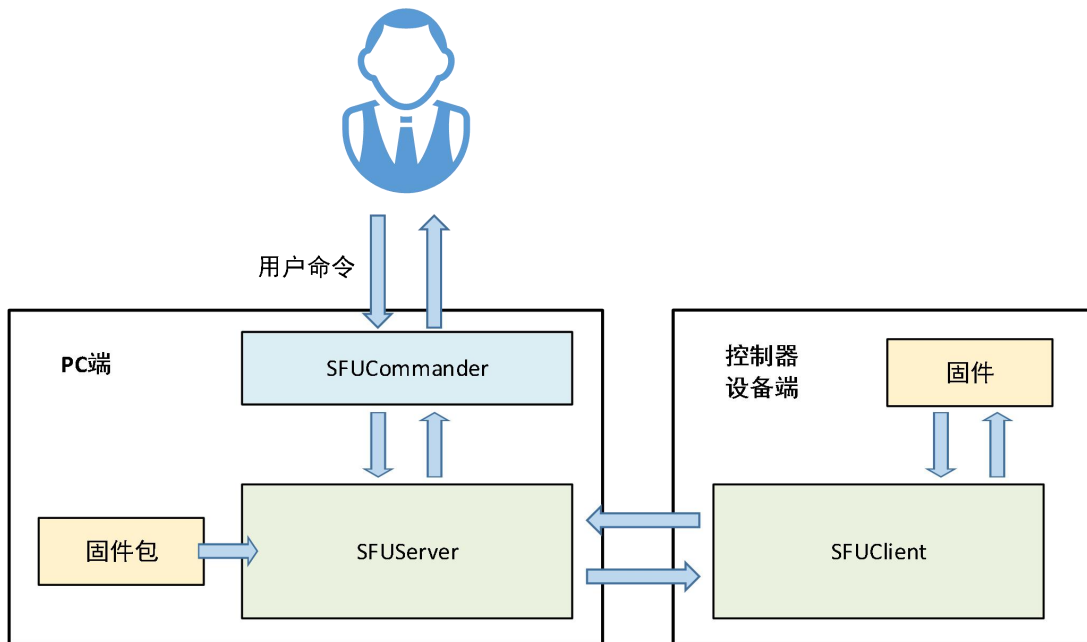


图 7-1 ROBOT SFU 系统组成

## 7.2 安装 SFU 固件升级工具

下载、解压 SFU 软件包：**probot\_sfu\_tool\_v1.0.0.zip**。检查软件包是否完整，完整的软件包应包含：

```
+ probot_sfu_tool_v1.0.0
  + doc
    * um304-probot-sfu-tool.pdf
  + Firmware
    * userConfig.json
  + SFUCommander
    * SFUCommander
  + SFUSever
    * SFUSever          + 文件夹
                        * 文件
```

## 7.3 使用

### 注意

(1) 请严格按照本说明使用 SFU 工具对控制器进行固件升级，否则可能导致无法开机、控制器无法启动等问题。

(2) 请勿对正在工作状态的控制器（如正在控制机械臂运动）执行固件升级操作。升级过程可能触发控制器重启，极可能引发危险！

(3) 在固件升级过程中，请确保控制器的供电正常，并且保持与 PC 的网络连接（不掉电、不断网），否则可能导致升级过程故障而无法开机。

### 7.3.1 准备固件包

下载固件包，如 **robcell-a1-anno-v1\_0\_0.img**，放置到 Firmware 文件夹下。

### 7.3.2 修改配置文件

文件路径：**Firmware/userConfig.json**

内容如下：

```
{
```

```
"path": "../Firmware/",  
"firmware": "robcell-a1-anno-v1_0_0.img",  
}
```

字段释义：

- (1) path - 固件包在 PC 端的存储位置，建议使用相对路径。
- (2) firmware - 固件包名。

### 注意

请确保配置文件内容符合 JSON 格式，以上所有字段的字符串长度不能超过 40 个字符。

### 7.3.3 连接控制器

- (1) 使用网线连接 PC 和控制器（如 ROBCELL）的网口。
- (2) 控制器上电运行，等待系统启动完成。
- (3) 修改 PC 的 IP 配置为 192.168.2.xxx，如 192.168.2.10，控制器默认为静态 IP：

192.168.2.123，应保证两者 IP 设置不相同。

- (4) 测试网络是否连通。在 Ubuntu 下执行命令：

```
$ ping 192.168.2.123
```

### 7.3.4 升级固件

- (1) 运行 SFUCommander 程序，进入 SFU 命令行交互界面。在 SFUCommander 目录下执行命令：

```
$ ./SFUCommander
```

SFUCommander 启动后将自动启动 SFUServer。设置 IP 和 Port，输入控制器的 IP 192.168.2.123，Port 使用默认值，即直接回车键跳过：

```
*****
Welcome to use SFU(System Firmware Update)-Commander.
Powered by PS-Micro.
Version-v1.0.0. Date-2019/4/3.
*****
Notice!!! Please use the 'quit' command to exit instead of 'Ctrl+C'.

SFU-Commander: Start OK.
Input SFU-Client's IP and Port.
Enter empty to use default.
ip: 192.168.2.123
port:
SFU-Commander: Set SFU-Client's IP and Port[192.168.2.123:6666].
SFU-Commander: SFU-Server connected.
SFU-Server: Start OK.
<<<
```

(2) 在 SFUCommande 命令行中输入 conn 命令，连接控制器。界面将提示连接成功，并输出控制器的软硬件信息，若无提示则说明未连接成功。

```
>>> conn
SFU-Server:
{
  "sfu_version": "v1.0.0",
  "platform": {
    "controller": "robcell_a1",
    "board": "zturn+myir_expanded",
    "robot": "arm_anno",
  },
  "version": {
    "name": "robcell-a1-anno-v1_0_0",
    "major_num": 1,
    "minor_num": 0,
    "revision_num": 0,
  },
  "image": {
    "BOOT.BIN": 1,
    "devicetree.dtb": 1,
    "uImage": 1,
    "uramdisk.image.gz": 1,
  },
}
SFU-Server: Connect to SFU-Client OK.
```

(3) 在 SFUCommander 命令行中输入 `update` 命令，启动固件升级。正常情况下 11MB 大小的固件大约需要 13s，请耐心等待，直到提示升级结果。**若升级成功，将自动重启控制器！**

```
>>> update
SFU-Server: User configurations:
SFU-Server: { "path": "../Firmware/", "firmware": "robcell-a1-anno-v1_0_0.img", "reboot": 0 }
SFU-Server: Firmware info:
{ "sfu_version": "v1.0.0", "platform": { "controller": "robcell_a1", "board": "zturn+myir_expanded", "robot": "arm_anno" }, "version": { "name": "robcell-a1-anno-v1_0_0", "major_num": 1, "minor_num": 0, "revision_num": 0 }, "image": { "BOOT.BIN": 1, "devicetree.dtb": 1, "ulmage": 1, "uramdisk.image.gz": 1 }
SFU-Server: Send Image file[robcell-a1-anno-v1_0_0.img] OK.
SFU-Server: Firmware update successfully in 13s.
```

输出信息包含用户配置信息、所使用的固件包信息，以及最终的更新结果。

### 7.3.5 退出

升级成功后，在 SFUCommander 命令行中输入 `quit` 命令，退出 SFUCommander:

```
>>> quit
SFU-Commander Exit!!!
```

#### 注意

请勿使用“Ctrl+C”直接退出 SFUCommander，否则将导致 SFUCommander 较长时间内都无法启动，可能需要重启 PC。

## 7.4 SFU 命令大全

SFUCommander 支持 以下命令：

命令	参数选项	功能简介
help	无	查看命令帮助
conn	无	向控制器（SFUClient）发起连接
disconn	无	断开与控制器（SFUClient）的连接
update	无	执行固件升级
quit	无	退出程序

## 7.5 常见问题

### 7.5.1 SFUCommander 无法启动

根据 7.3.4 小节，使用命令 ./SFUCommander 无法启动 SFUCommander，且无任何输出提示。可能是由于上次建立的网络连接尚未完全关闭，请等待 1 分钟左右时间后，再次尝试启动 SFUCommander。否则，重启 PC 后再尝试。

### 7.5.2 版本不兼容

(1) 若提示以下信息，表示所使用的 SFU 工具与控制器当前运行的 SFUClient 程序版本不兼容：

```
>>> update
SFU-Server: The version of SFU-Client running on this controller: v2.1.0, Please use the compatible SFU tool[v1.0.0 != v2.1.0].
```

请检查 SFU 工具版本，必须使用完全一致的版本号才能进行固件升级。如以上情况，应使用 **probot\_sfu\_tool\_v2.1.0.zip** SFU 工具包才能完成升级。

(2) 若提示以下信息，表示所使用的固件包与控制器版本不兼容：

```
>>> update
SFU-Server: The firmware is not compatible with this controller: [robcell_a1 != robcell_a2].
```

请检查所使用的控制器版本。如以上情况，应使用支持 **ROBCELL a2** 控制器的 **robcell-a2-xxx-xxx.img** 固件包才能完成升级。

(3) 若提示以下信息，表示所使用的固件包与控制器硬件版本不兼容：

```
>>> update
SFU-Server: The firmware is not compatible with this controller: [zturn+myir_expanded != zturn_lite].
```

请检查所使用控制器的硬件版本。如以上情况，应使用支持米尔 **zturn-lite** 硬件平台的 **robcell-a1-anno-xxx.img** 固件包才能完成升级。

(4) 若提示以下信息，表示所使用的固件包与控制器对接的机械臂版本不兼容：

```
>>> update
SFU-Server: The firmware is not compatible with this controller: [arm_anno != arm_annisen].
```

请检查所使用的机械臂版本。如以上情况，应使用支持安尼森机械臂的 **robcell-a1-annisen-xxx.img** 固件包才能完成升级。

### 7.5.3 无需升级

若提示以下信息，表示控制器当前运行的固件比使用的固件包版本更新，无需升级：

```
>>> update
```

```
SFU-Server: The firmware running on this controller is more fresher: [robcell-a1-anno-v1_0_0 <= robcell-a1-anno-v1_1_0]. You don't need to update firmware.
```

请检查所使用的固件包版本。如以上情况，应使用比 `robcell-a1-anno-v1_1_0.img` 更高版本的固件包进行升级，如 `robcell-a1-anno-v1_2_0.img`。

#### 7.5.4 找不到固件包

若提示以下信息，表示 PC（SFU Server）根据配置文件 `userConfig.json` 找不到固件包，无法进行固件升级：

```
>>> update
```

```
SFU-Server: Firmware [robcell-a1-anno-v1_0_0.img] not found.
```

```
SFU-Server: Unzip compressed firmware failed.
```

请检查所使用的固件包的名称、存储路径等，并编辑修改配置文件 `userConfig.json`，修改方法参考见 7.3.2 小节。

# 8 常见故障诊断

---

## 8.1 无法开机

检查电源输入线缆是否已接好，电源插头是否插到有效市电网络。中国大陆电源电压 220V，中国台湾、日本、欧美等地区电源电压 110V。

电源开关是否处于开启状态（电源开关处于开启状态时，电源开关指示灯亮。）急停开关是否处于急停状态，如果是以上原因，可以自行诊断解决；否者，请及时和我们的维修部联系，请不要在不熟悉的情况下自己拆装机器。售后问题联系邮箱：[support@ps-micro.com](mailto:support@ps-micro.com)

## 8.2 机械臂无法回到原点

处理方法：如果在操作过程中，发现机械臂回不了原点，请先关闭电源，手动将机械臂转动到原点位置。然后再打开电源，重新设置原点位置，调试运转几次，查看该问题是否排除。如果多次出现该问题，请及时与我们的售后部联系。请不要在不熟悉的情况下自己擅自拆动机器。

## 8.3 机械臂抓不紧物体

机械臂如果出现抓不紧物体等，可能因为机械臂抓取物体超载或者机械臂使用时间长引起的零件磨损等问题。如果多次出现该问题，请及时和我们的维修部联系，请不要在不熟悉的情况下自己拆机器。

### 注意

如果不是因为被抓物体超重引起的，请找修理人员处理，不要自己私自处理，否则出现其他问题我们概不负责。

## 8.4 上位机关闭后无法连接控制器

上位机关闭后，如控制柜不重新启动，将会出现无法连接的问题，终端中有红色报警出现，请关闭上位机及电控柜电源，重新启动电控柜电源后，再启动上位机界面，即可连接。

## 9.1 参考资料

1. ROS Wiki - PROBOT: [http://wiki.ros.org/Robots/PROBOT\\_Anno](http://wiki.ros.org/Robots/PROBOT_Anno)
2. Github - PROBOT: [https://github.com/ps-micro/PROBOT\\_Anno](https://github.com/ps-micro/PROBOT_Anno)
3. MoveIt! Tutorials: [http://docs.ros.org/kinetic/api/moveit\\_tutorials/html/index.html](http://docs.ros.org/kinetic/api/moveit_tutorials/html/index.html)
4. 《ROS 机器人开发实践》，胡春旭编著，机械工业出版社
5. 《ROBOT SFU 固件升级工具用户手册》

## 9.2 联系方式

邮箱: [sales@robotanno.com](mailto:sales@robotanno.com)

网站: [www.robotanno.com](http://www.robotanno.com)

[www.gyh.ai](http://www.gyh.ai)

