

Baxter™

研究版•操作手册



作者	版本	编订日期
(胡博士)	1.0	2016/10/1

内容提要

Baxter 机器人是总部位于波士顿的美国 **Rethink Robotics** 公司推出的一款新型协作型机器人。**Baxter** 配备了基于 **ROS** (**Robot Operating System**) 的软件开发套件 (**SDK**)，是一个安全、经济且强大的平台。

目前全球大多数实验室和学校都配有 **Baxter**，该机器人广泛应用于机械手臂运动规划、双臂柔顺协调控制、机器视觉、人机交互等领域的科研和教学活动。

此操作手册共分三个部分，分别是 **Baxter** 设置，工作站设置，运行示例。

Baxter 设置部分包括：硬件配置、工作空间选择、夹持器安装等。

工作站设置包括：**Ubuntu** 及 **ROS** 安装、依赖软件包安装、**SDK** 安装等。

运行示例包括：使能机器人、运行示例程序。

通过使用该手册，初学者能够快速熟悉 **Baxter** 的基本控制和工作流程，对其软硬件有一定初步认识。

目录

内容提要.....	2
1 Baxter 设置.....	4
1.1 所需硬件.....	4
1.2 选择合适的工作空间.....	4
1.3 Baxter 安装.....	8
1.4 安装夹持器.....	10
1.4.1 安装电动夹持器.....	10
1.4.2 安装气动夹持器.....	12
1.5 连接急停开关及电源.....	13
1.6 打开电源.....	14
2 工作站设置.....	14
2.1 安装 Ubuntu.....	14
2.2 安装 ROS.....	19
2.3 安装 SDK.....	21
3 运行 Hello Baxter 示例.....	23

1 Baxter 设置

1.1 所需硬件

- Baxter 科研版机器人本体
- 1/2 英寸扳手
- 17mm 扳手
- 27mm 扳手（用于底座安装）
- 平头螺丝刀
- 电动平行夹持器或启动夹持器
- Baxter 底座
- 内六角扳手一套
- USB 键盘
- 路由器及网线

1.2 选择合适的工作空间

Baxter 机器人在工作时应保留足够的空间以免机械臂碰到障碍物停止运动，Baxter 工作空间分布参考图 1.1-图 1.4。

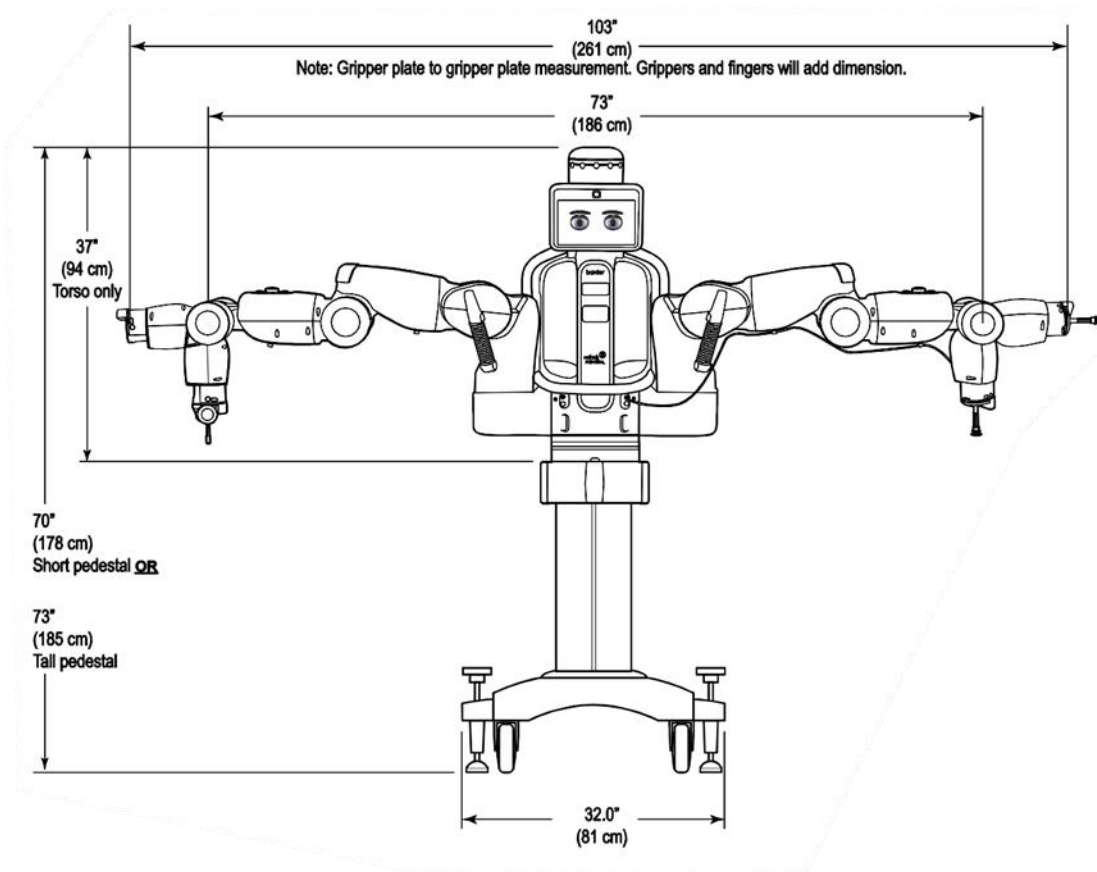


图 1.1 Baxter 正视图

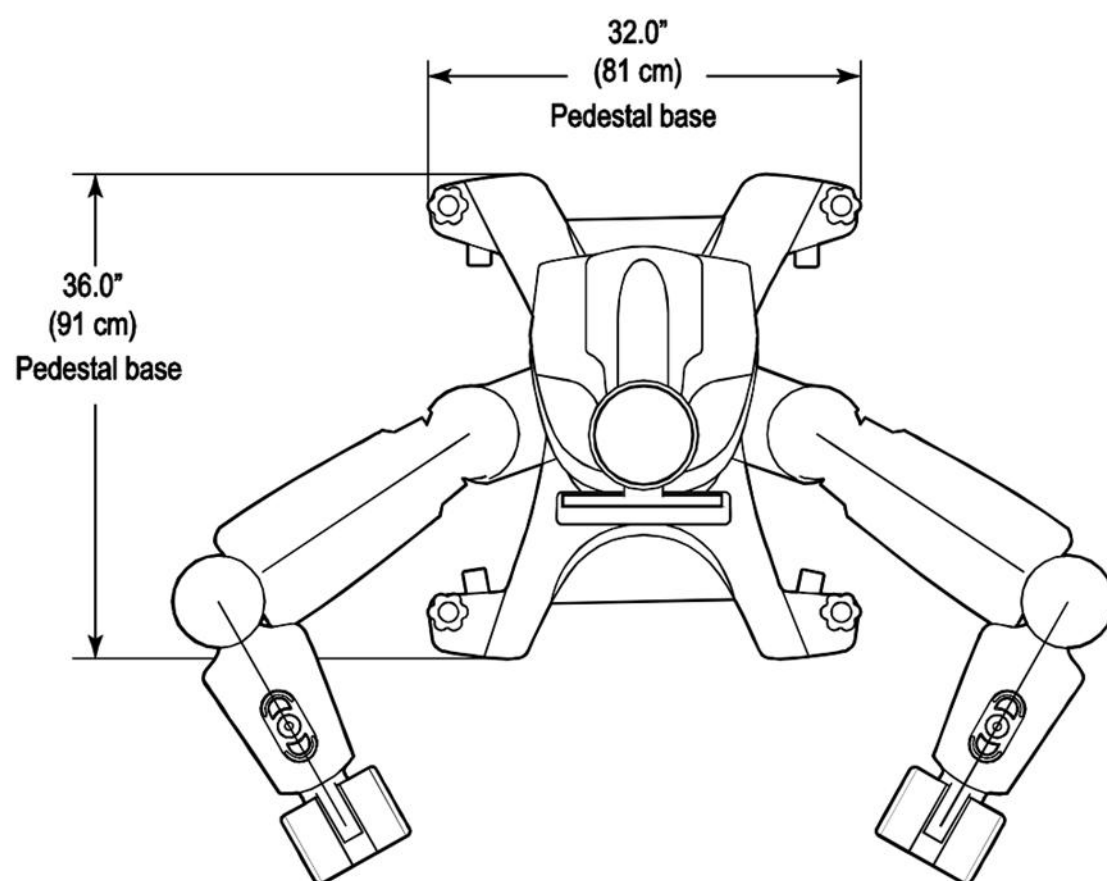


图 1.2 Baxter 俯视图

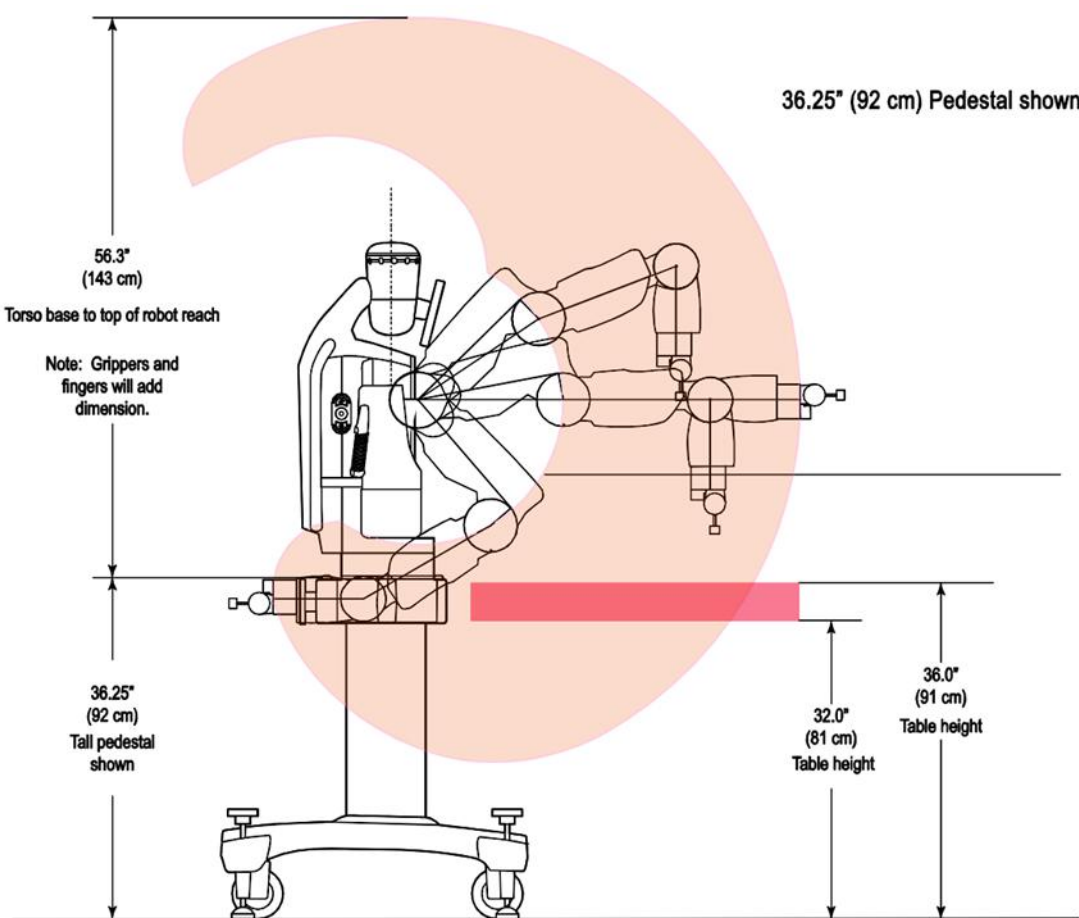


图 1.3 Baxter 侧视图

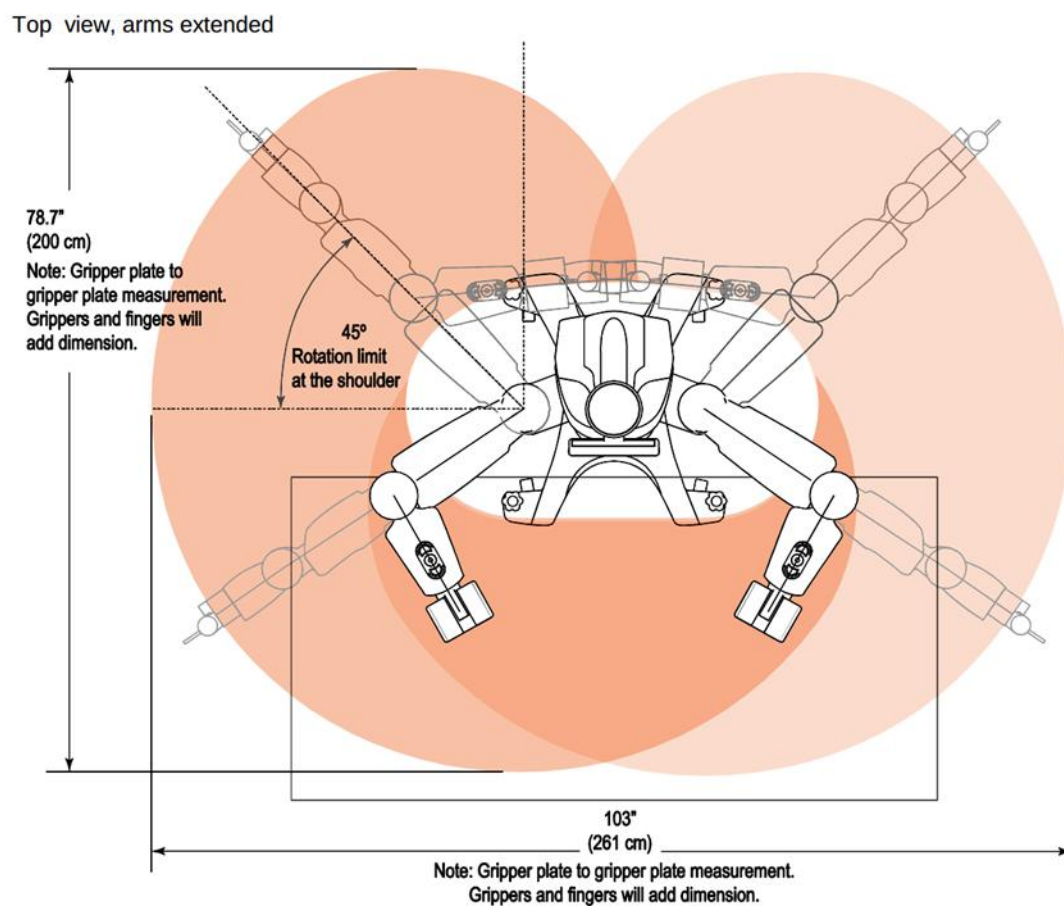


图 1.4 Baxter 机械臂云动范围

1.3 Baxter 安装

首先参考图 1.5 选择底座安装高度。

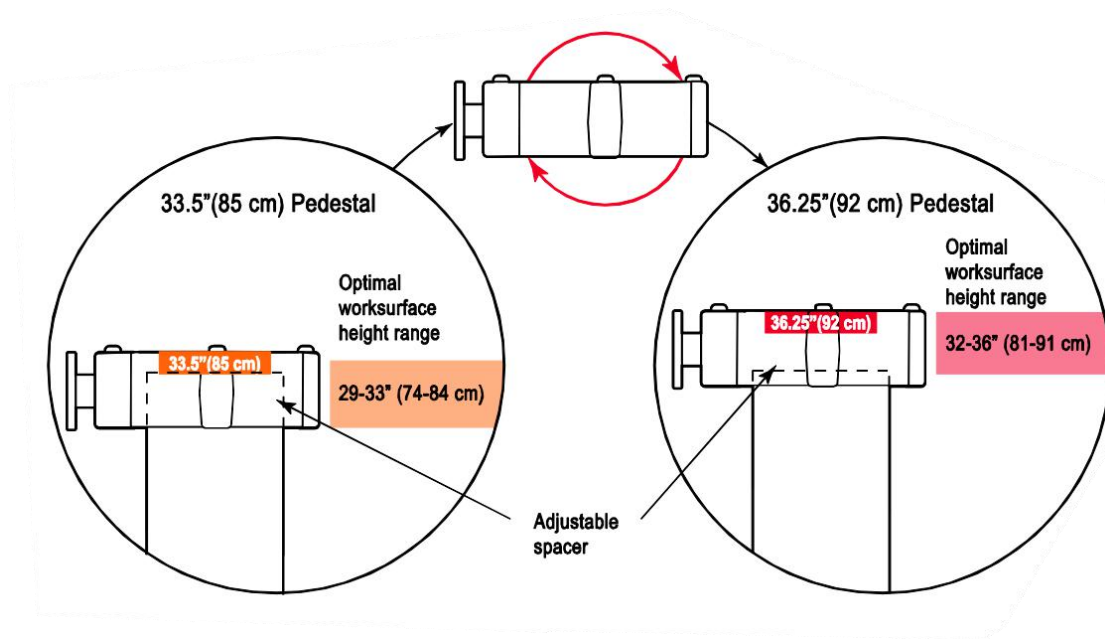


图 1.5 Baxter 底座安装高度

然后将 Baxter 本体吊装到底座上，如图 1.6 所示。

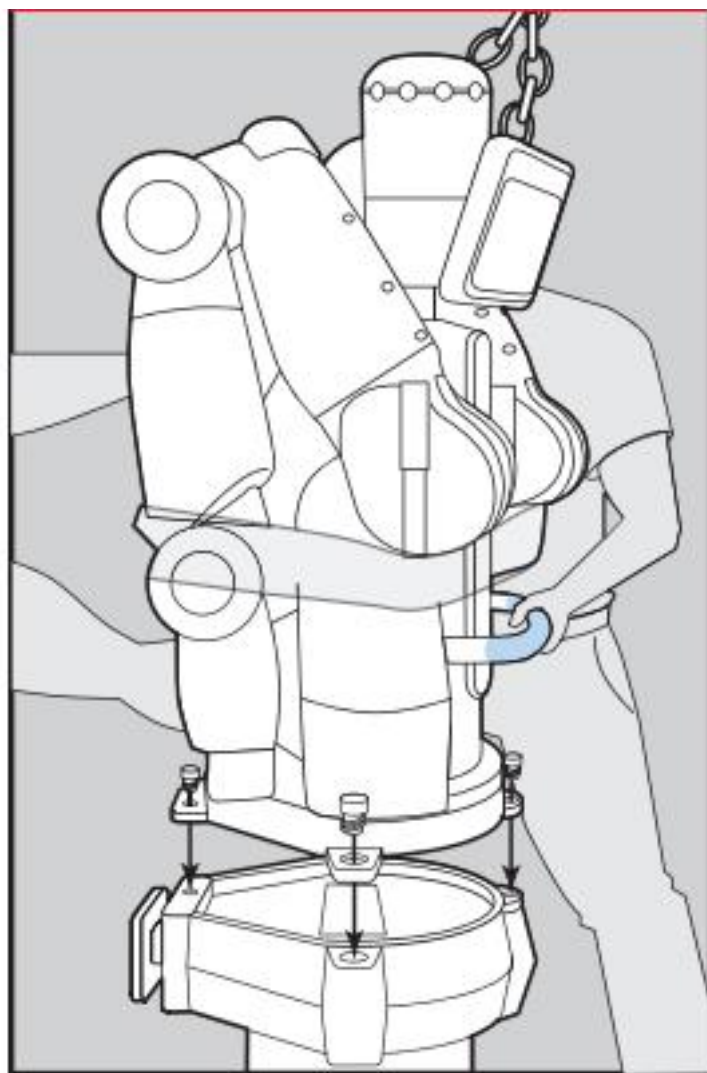


图 1.6 吊装 Baxter

1.4 安装夹持器

1.4.1 安装电动夹持器

参考图 1.7-图 1.9 所示步骤。

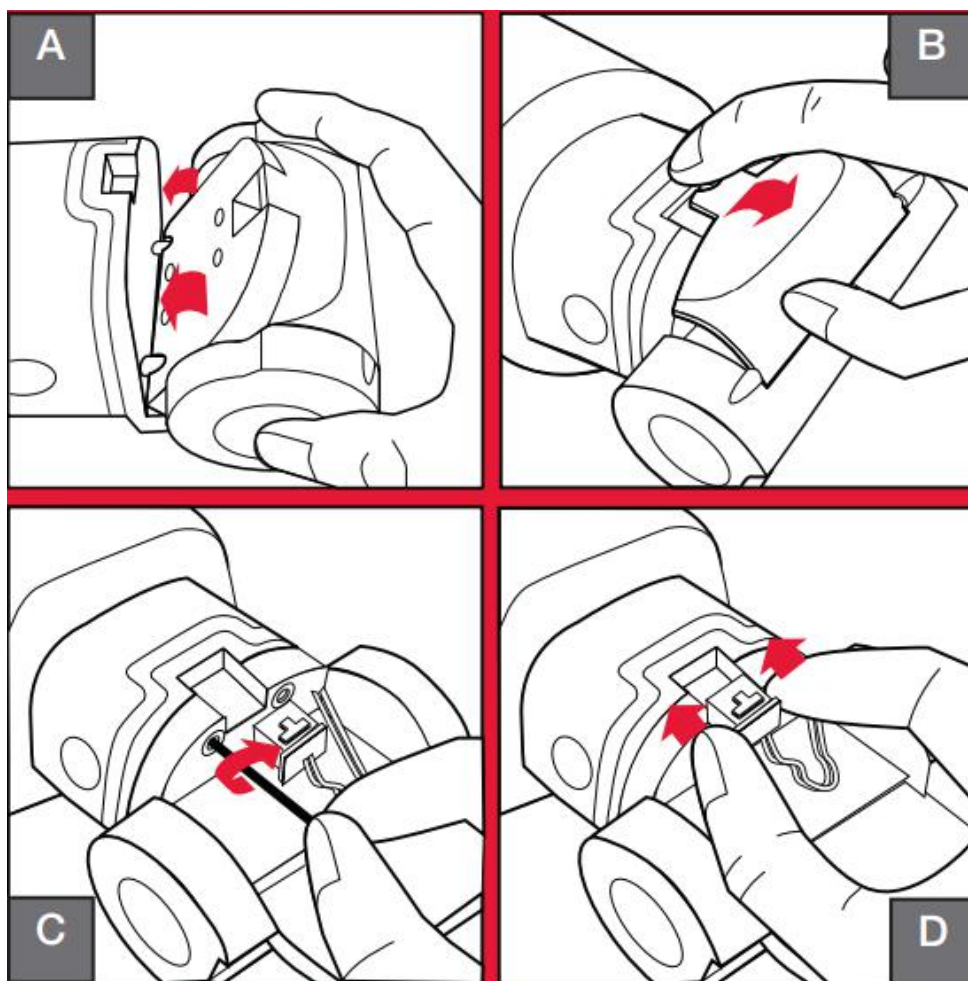


图 1.7 安装电动夹持器基座

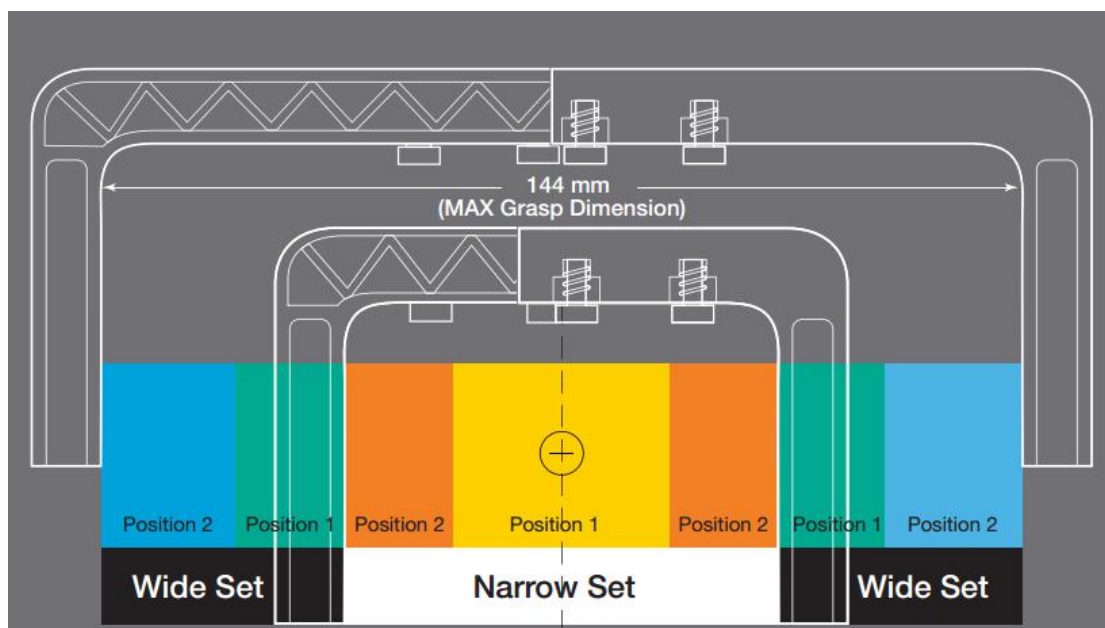


图 1.8 选择不同开度的手指

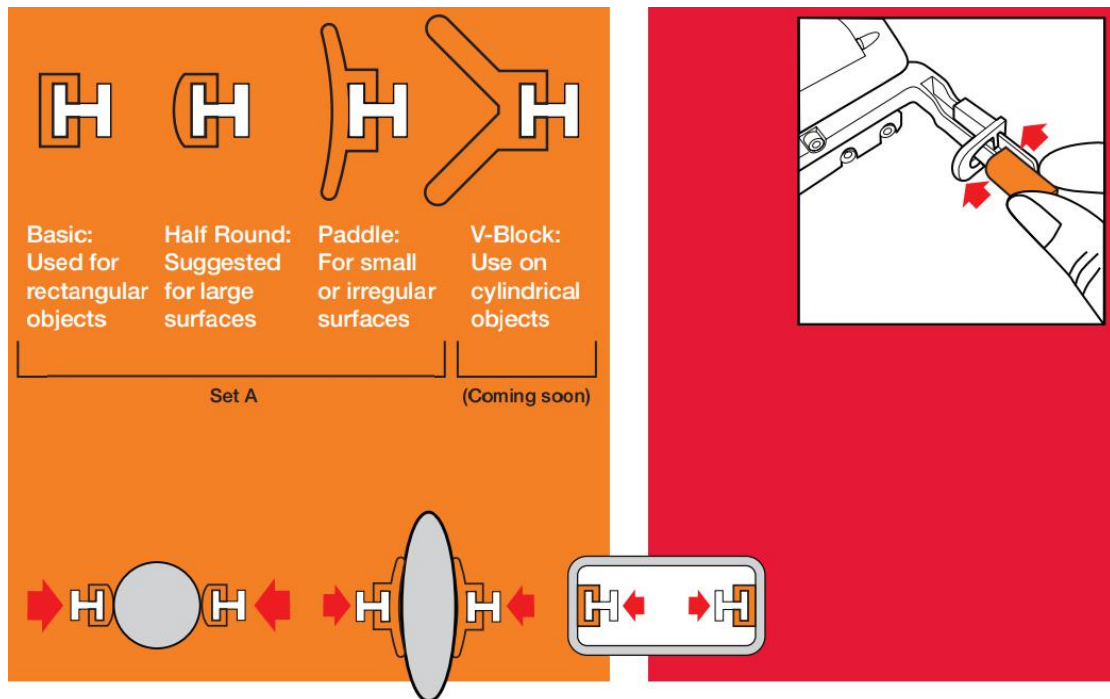


图 1.9 选择指端

1.4.2 安装气动夹持器

参考图 1.10-图 1.3 所示步骤。

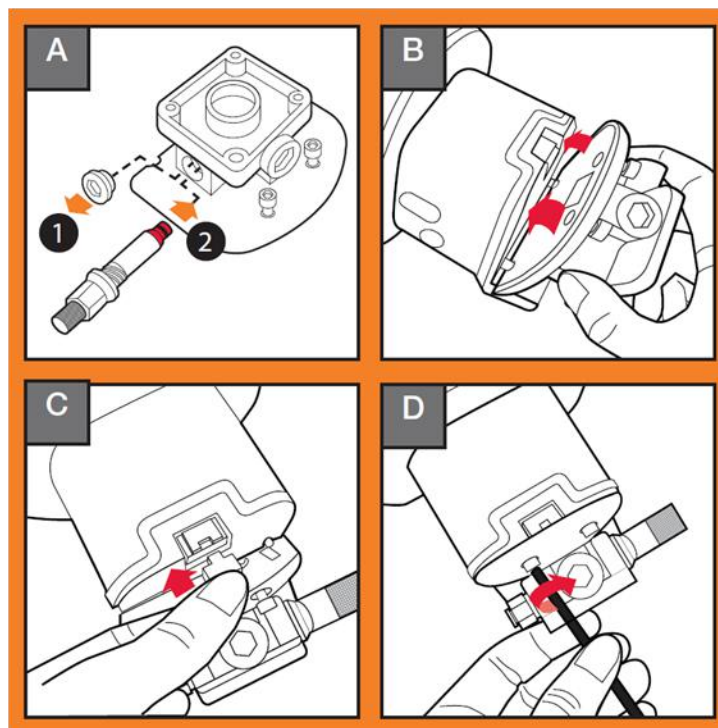


图 1.10 安装气动夹持器基座

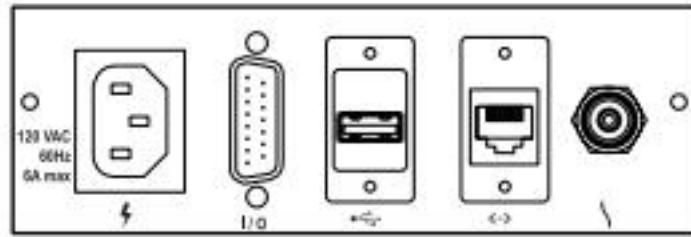


图 1.14 将急停开关连接至 I/O 端口

1.6 打开电源

至此，Baxter 机器人硬件配置已经完成，打开背部开关后，约 3 分钟后会显示图 1.15 的欢迎界面。



图 1.15 Baxter 科研版开机欢迎界面

2 工作站设置

2.1 安装 Ubuntu

此手册以 Ubuntu 14.04 为例。

A) 下载 Ubuntu 镜像：

<http://releases.ubuntu.com/trusty/ubuntu-14.04.3-desktop-amd64.iso>

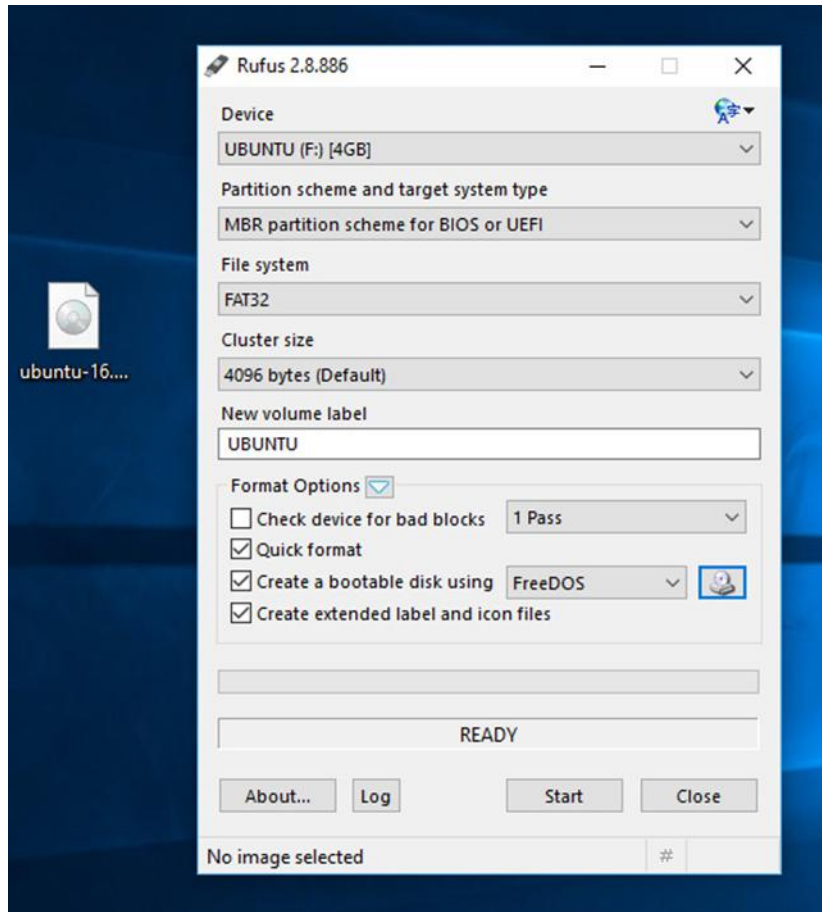
B) 创建 USB 启动盘，可使用 Unetboot 或 Rufus：

Unetboot 下载链接：<http://unetbootin.github.io/>

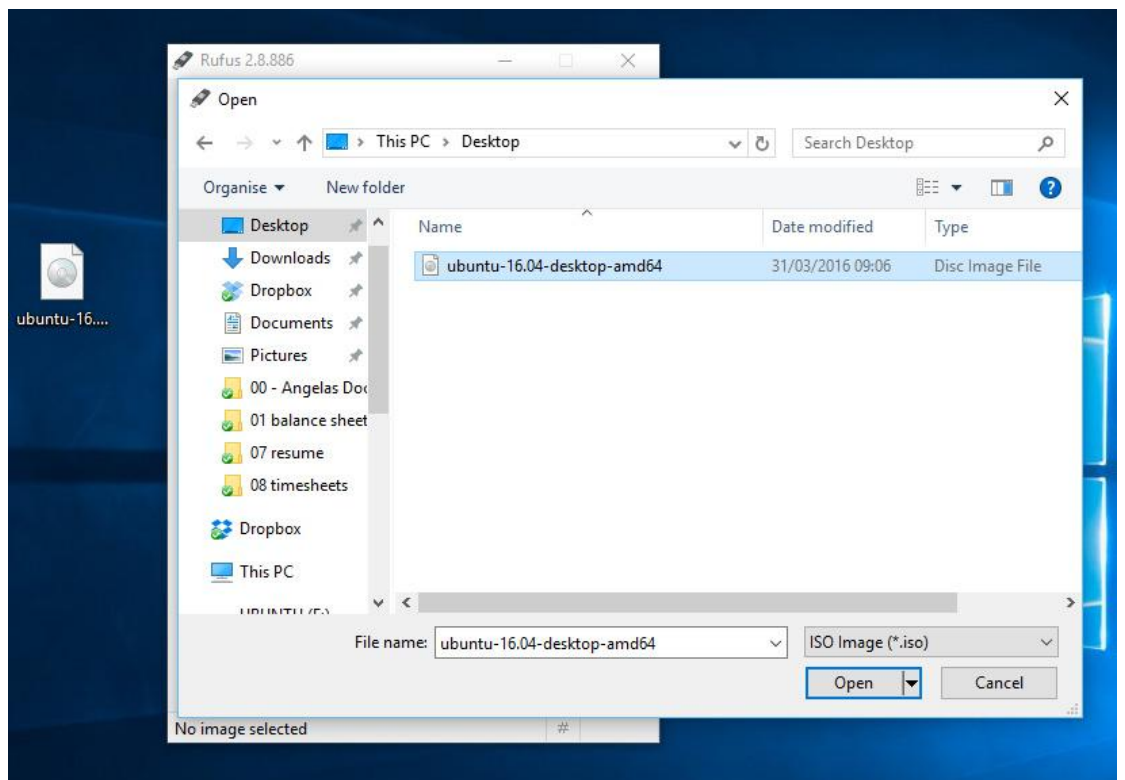
Rufus 下载链接：<https://rufus.akeo.ie/>

下面以 Rufus 为例介绍如何制作 USB 启动盘。

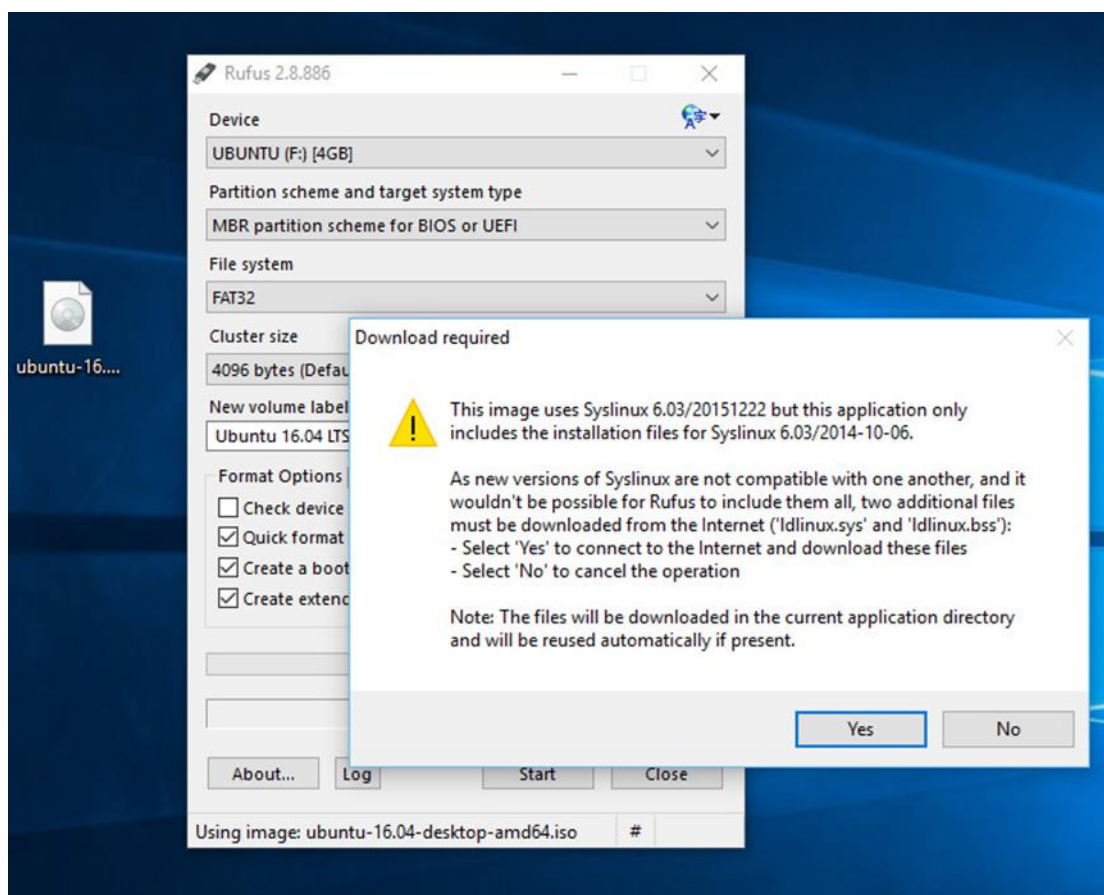
1、打开 Rufus，选择 USB 设备；



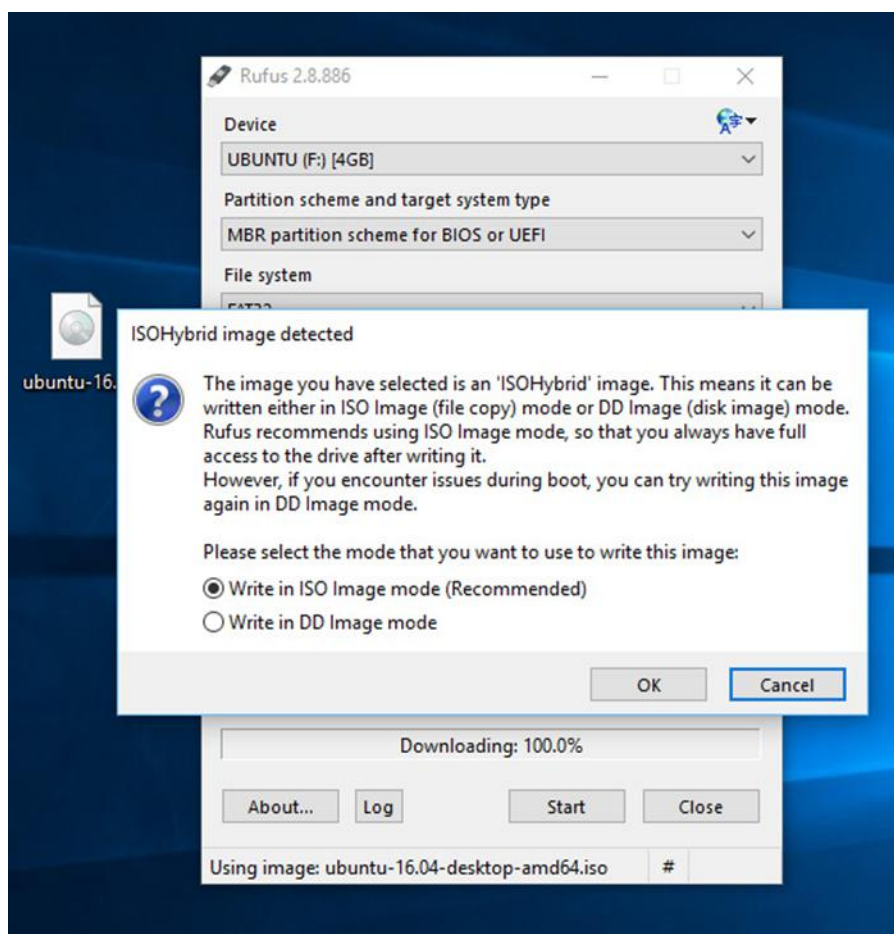
2、选择下载的 Ubuntu 镜像；



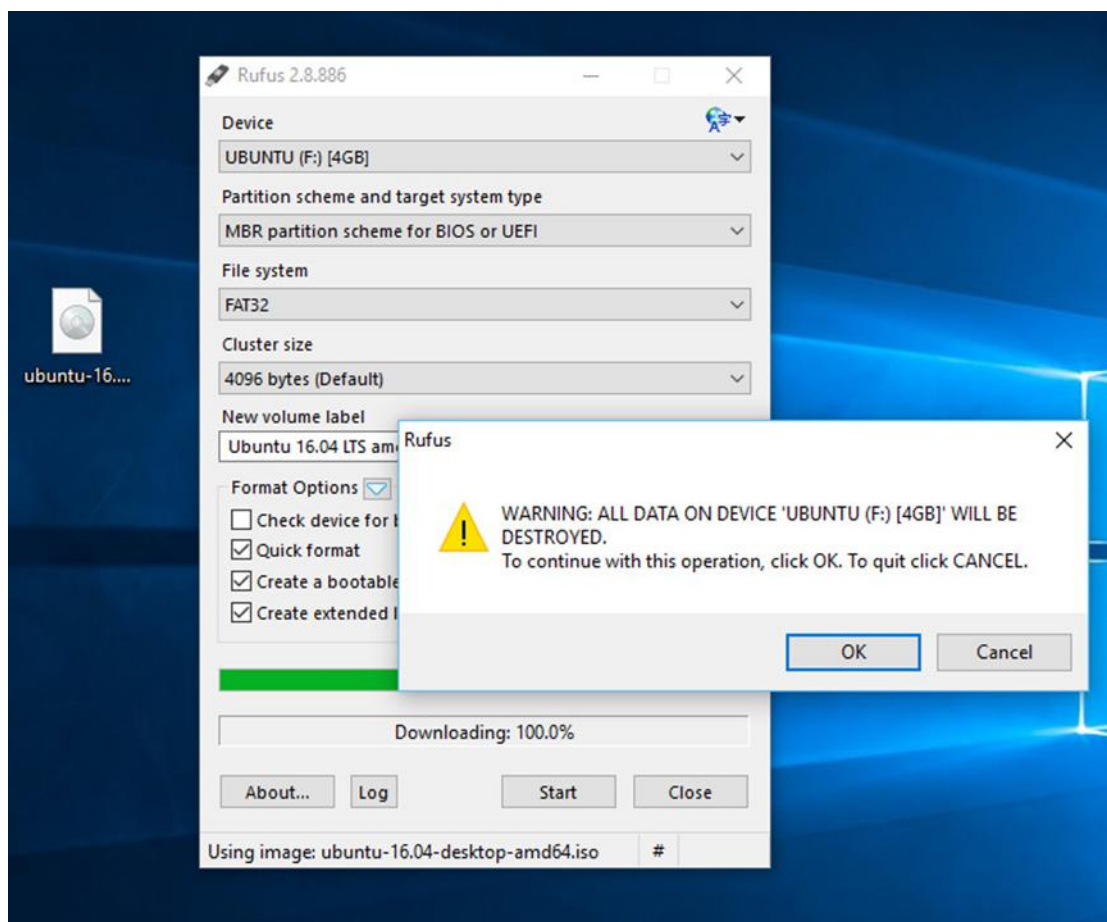
3、点击 “Yes” 下载 Syslinux 软件；



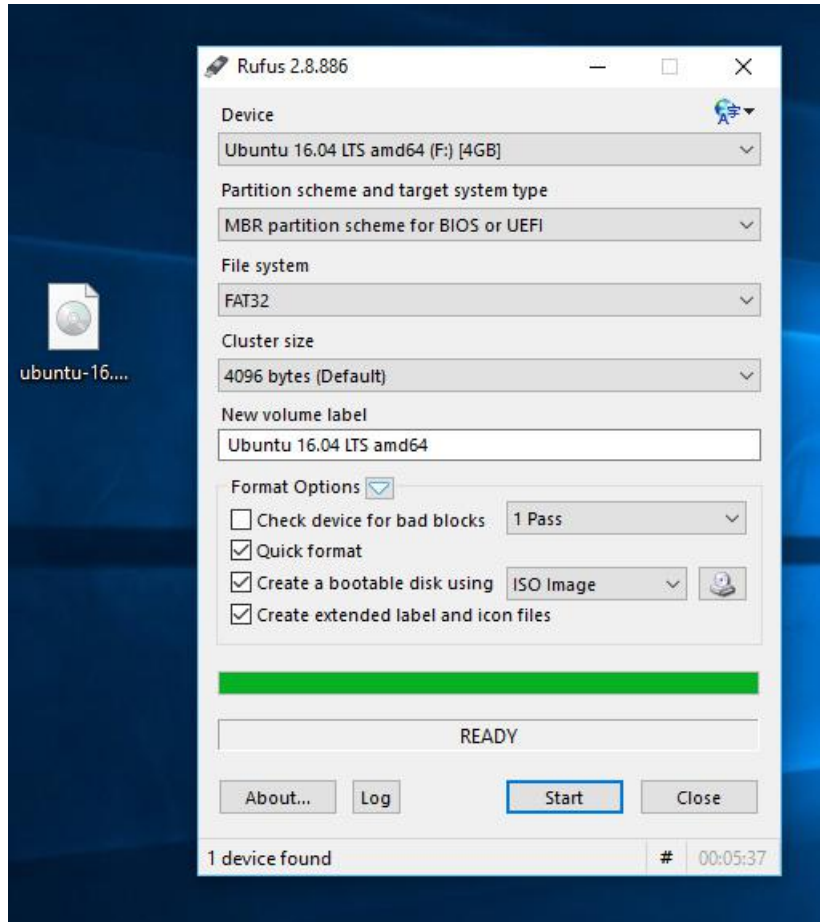
4、点击“OK”以 ISO 方式写入；



5、确认 USB 设备；



6、写入完成后，重启电脑，选择从 U 盘启动。



2.2 安装 ROS

此手册以 ROS Indigo 为例。

1、配置 Ubuntu 软件仓库

配置你的 Ubuntu 软件仓库(repositories) 以允许 "restricted"、"universe" 和 "multiverse"这三种安装模式。

2、添加 sources.list

配置你的电脑使其能够安装来自 packages.ros.org 的软件。 ROS Indigo 仅支持 Saucy (13.10) 和 Trusty (14.04)。

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

3、添加 keys

- `sudo apt-key adv --keyserver hkp://pool.sks-keyservers.net --recv-key 0xB01FA116`

4、安装

首先，确保你的 **Debian** 软件包索引是最新的：

- `sudo apt-get update`

ROS 中有很多各种函数库和工具，我们为你提供了四种默认安装方式，你也可以单独安装某个指定软件包。

- **桌面完整版安装：（推荐）** 包含 ROS、[rqt](#)、[rviz](#)、通用机器人函数库、2D/3D 仿真器、导航以及 2D/3D 感知功能。

在 **Trusty** 中 **Indigo** 默认搭配使用 **Gazebo2**。

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

桌面版安装： 包含 ROS、[rqt](#)、[rviz](#) 以及通用机器人函数库。

- `sudo apt-get install ros-indigo-desktop`

基础版安装： 包含 ROS 核心软件包、构建工具以及通信相关的程序库，无 GUI 工具。

- `sudo apt-get install ros-indigo-ros-base`

单个软件包安装： 你也可以安装某个指定的 ROS 软件包（使用软件包名称替换掉下面的 **PACKAGE**）：

- `sudo apt-get install ros-indigo-PACKAGE`

例如：

```
sudo apt-get install ros-indigo-slam-gmapping
```

要查找可用软件包，请运行：

```
apt-cache search ros-indigo
```

5、初始化 rosdep

在开始使用 ROS 之前你还需要初始化 rosdep。rosdep 可以方便在你需要编译某些源码的时候为其安装一些系统依赖，同时也是某些 ROS 核心功能组件所必需用到的工具。

```
sudo rosdep init
```

```
rosdep update
```

6、环境设置

如果每次打开一个新的终端时 ROS 环境变量都能够自动配置好（即添加到 `bash` 会话中），那将会方便得多：

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

如果你安装有多个 ROS 版本，`~/.bashrc` 必须只能 `source` 你当前使用版本所对应的 `setup.bash`。

如果你只想改变当前终端下的环境变量，你可以执行命令：

```
source /opt/ros/indigo/setup.bash
```

7、安装 `roscpp`

[roscpp](#) 是 ROS 中一个独立分开的常用命令行工具，它可以方便让你通过一条命令就可以给某个 ROS 软件包下载很多源码树。

要在 `ubuntu` 上安装这个工具，请运行：

```
sudo apt-get install python-roscpp
```

8、Build farm 状态

你所安装的各种软件包都是通过 [ROS build farm](#) 来编译构建的。你可以在[这里](#)查看各种独立软件包的编译状态。

2.3 安装 SDK

安装完 `Ubuntu 14.04` 和 `ROS Indigo` 之后，我们将在工作站上安装 `Baxter SDK`。

1、创建 ROS 工作空间

```
$ mkdir -p ~/ros_ws/src
```

2、编译安装

```
$ source /opt/ros/indigo/setup.bash  
$ cd ~/ros_ws  
$ catkin_make  
$ catkin_make install
```

3、安装 SDK 依赖

```
$ sudo apt-get update

$ sudo apt-get install git-core python-argparse python-wstool python-vcstools python-r
osdep ros-indigo-control-msgs ros-indigo-joystick-drivers
```

4、安装 Baxter 科研版 SDK

```
$ cd ~/ros_ws/src
$ wstool init .
$ wstool merge https://raw.githubusercontent.com/RethinkRobotics/baxter/master/baxter_sdk.rosinstall
$ wstool update

$ source /opt/ros/indigo/setup.bash

$ cd ~/ros_ws
$ catkin_make

$ catkin make install
```

5、下载并修改 baxter.sh，配置 Baxter 通讯

```
$ wget https://github.com/RethinkRobotics/baxter/raw/master/baxter.sh
$ chmod u+x baxter.sh
$ cd ~/ros_ws
$ gedit baxter.sh
# 星号内为要修改的内容
**baxter_hostname="baxter_hostname.local"**
**your_ip="192.168.XXX.XXX"**
***ros_version="indigo"***
```

6、保存 baxter.sh，初始化 SDK 运行环境

```
$ cd ~/ros_ws
$ . baxter.sh.
```

7、验证 SDK 安装

```
$ env | grep ROS

# ROS_MASTER_URI - 应是机器人的主机名

# ROS_IP - 应是工作站 IP

# ROS_HOSTNAME - 与 ROS_IP 选一即可，指定工作站主机名
```

3 运行 Hello Baxter 示例

1、设置 ROS 及 Baxter 运行环境

```
$ cd ~/ros_ws

$ source /opt/ros/indigo/setup.bash

$ catkin_make

# Source baxter.sh script

$ . baxter.sh
```

2、验证网络连接，查看 baxter topics

```
# 确定 ROS 主节点 URI

$ env | grep ROS_MASTER_URI

# Ping ROS 主节点

$ ping <our ROS Master>

#如

$ ping 011303P0017.local

# baxter topic 列表
```

3、通过 SSH 从工作站登录 baxter 机器人

```
$ $ ssh ruser@<our ROS Master>
```

```
# 密码: rethink

# 如:

$ ssh ruser@011303P0017.local

#现在已经登录到机器人, 请从机器人 ping 工作站

ruser@p99 ~ $ ping <ROS_IP/ROS_HOSTNAME>

# 如

ruser@p99 ~ $ ping 192.168.101.99

# 或使用 ROS_HOSTNAME

ruser@p99 ~ $ ping yoda

$ . baxter.sh.
```

4、使能机器人

```
$ cd ~/ros_ws

$ . baxter.sh.

$ rosrun baxter_tools enable_robot.py -e
```

此时机器人已经处于工作状态, 按住手腕处的薄膜按钮, 机器人将进入零重力模式, 您可以自由拖动机械臂。

5、运行示例程序

```
$ rosrun baxter_examples joint_velocity_wobbler.py
```

执行上述命令后, 机器臂首先移动到中间位置, 进入速度控制模式, 每个关节进行随机正弦运动。

6、交互式编程

打开 Python 命令窗口, 可逐行输入指令。

```
$ python

# 导入必要的 python 模块

# rospy - ROS Python API

>>> import rospy

# baxter_interface - Baxter Python API

>>> import baxter_interface

# initialize our ROS node, registering it with the Master
```



```

>>> rospy.init_node('Hello_Baxter')

# create an instance of baxter_interface's Limb class

>>> limb = baxter_interface.Limb('right')

# get the right limb's current joint angles

>>> angles = limb.joint_angles()

# print the current joint angles

>>> print angles

# reassign new joint angles (all zeros) which we will later command to the limb

>>> angles['right_s0']=0.0

>>> angles['right_s1']=0.0

>>> angles['right_e0']=0.0

>>> angles['right_e1']=0.0

>>> angles['right_w0']=0.0

>>> angles['right_w1']=0.0

>>> angles['right_w2']=0.0

# print the joint angle command

>>> print angles

# move the right arm to those joint angles

>>> limb.move_to_joint_positions(angles)

# Baxter wants to say hello, let's wave the arm

# store the first wave position

>>> wave_1 = {'right_s0': -0.459, 'right_s1': -0.202, 'right_e0': 1.807, 'right_e1': 1.714,
              'right_w0': -0.906, 'right_w1': -1.545, 'right_w2': -0.276}

# store the second wave position

>>> wave_2 = {'right_s0': -0.395, 'right_s1': -0.202, 'right_e0': 1.831, 'right_e1': 1.981,
              'right_w0': -1.979, 'right_w1': -1.100, 'right_w2': -0.448}

# wave three times

>>> for _move in range(3):
...     limb.move_to_joint_positions(wave_1)
...     limb.move_to_joint_positions(wave_2)

```

```
# quit  
>>> quit()
```

至此您已经基本熟悉了 **Baxter** 的工作流程并能够使用命令控制 **Baxter** 的基本运动了。关于 **Baxter** 工作原理的细节和应用案例，请参考配套的 **Baxter** 教材。