

机器人

JIQIREN CAOZUO
XITONG JICHU
YU YINGYONG
(DI-ERBAN)



机器人及
人工智能类
创新教材

操作系统(ROS)基础与应用

总主编 冷晓琨

主 编 叶剑锋 赵 魁 杨 驰

(第二版)



AI



哈尔滨工业大学出版社
HARBIN INSTITUTE OF TECHNOLOGY PRESS

机器人及人工智能类创新教材

机器人建模与仿真（第二版）

协作机器人基础（第二版）

智能人机交互技术（第二版）

◆ **机器人操作系统(ROS)基础与应用（第二版）**

机器人运动控制（第二版）

机器人驱动与控制技术（第二版）

机器学习（第二版）

图像处理与计算机视觉（第二版）

机器人编程与实践



策划编辑 李艳文 范业婷

责任编辑 李佳莹

定价:68.00元

机器人

JIQIREN CAOZUO
XITONG JICHU
YU YINGYONG
(DI-ERBAN)



机器人及
人工智能类
创新教材

操作系统(ROS)基础与应用

(第二版)

总主编 冷晓琨

主 编 叶剑锋 赵 魁 杨 驰

副主编 李燕梅 徐 波 李 卓 刘华锋 申云成

编 者 姜 彬

AI



哈尔滨工业大学出版社
HARBIN INSTITUTE OF TECHNOLOGY PRESS



内容简介

机器人操作系统（RobotOperatingSystem，ROS）已成为机器人领域的主流软件平台与事实标准。本书作为一本ROS初学者的入门教程，从工程实际和应用的角度出发，系统地介绍了ROS的基本概念与编程开发方法，内容深入浅出，通过精心设计的示例，可以帮助ROS零基础的读者在充分了解和掌握ROS编程开发方法的基础上，高效地使用ROS软件平台进行机器人的编程开发工作。

本书可以作为高等职业院校机器人技术、机电一体化、智能制造、人工智能等相关专业的教材，也可以作为其他类型院校相关专业的师生或从事相关工作的工程技术人员的参考用书。

图书在版编目(CIP)数据

机器人操作系统（Ros）基础与应用 / 叶剑锋, 赵魁, 杨驰主编. -- 2版. -- 哈尔滨 : 哈尔滨工业大学出版社, 2025. 2. -- ISBN 978-7-5767-1968-0

I. TP242

中国国家版本馆CIP数据核字第2025CC8844号

机器人操作系统（ROS）基础与应用（第二版）

JIQIREN CAOZUO XITONG (ROS) JICHU YU YINGYONG(DI-ERBAN)

策划编辑 李艳文 范业婷

责任编辑 李佳莹

出版发行 哈尔滨工业大学出版社

社 址 哈尔滨市南岗区复华四道街10号 邮编 150006

传 真 0451-86414749

网 址 <http://hitpress.hit.edu.cn>

印 刷 天津市蓟县宏图印务有限公司

开 本 787毫米×1092毫米 1/16 印张15.5 字数310千字

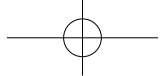
版 次 2023年7月第1版 2025年2月第2版 2025年2月第1次印刷

书 号 ISBN 978-7-5767-1968-0

定 价 68.00 元

（如因印装质量问题影响阅读，我社负责调换）





Preface 前言

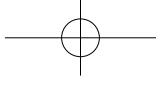


在数字化与智能化浪潮重塑全球制造业格局的今天，机器人技术已成为推动产业升级的重要引擎。《中国制造 2025》战略、工信部《“十四五”机器人产业发展规划》以及教育部《关于深化现代职业教育体系建设改革的意见》等政策，明确提出加快培养适应智能制造领域的高素质技术技能人才。与此同时，国家机器人安全标准体系的完善（如《国家机器人安全标准体系建设指南（2023 版）》）为技术研发与产业化应用提供了重要保障。在这一背景下，《机器人操作系统（ROS）基础与应用（第二版）》教材应运而生，旨在系统化培养掌握 ROS 核心技术、兼具职业素养与创新能力的技能人才。

一、编写理念与目标定位

本书严格遵循教育部《职业院校教材管理办法》要求，以“产教融合、德技并修”为核心指导思想，聚焦立德树人根本任务，构建知识传授—技能培养—职业素养三位一体的培养体系。通过项目引领、任务驱动的教学设计，将 ROS 基础理论与工业应用场景紧密结合，帮助学生掌握从系统开发到工程落地的完整技术链条。





思 政 融 合

铸 魂 育 人

产 教 协 同

标 准 对 接

岗 课 赛 证 融 通

赋 能 职 业 发 展

二、教材内容架构

全书以“基础认知→技能进阶→项目实战”为主线，分为三个篇章，层层递进，知行合一。

基础理论篇：从ROS的诞生背景、核心架构（节点、话题、服务、参数服务器）到数学基础与建模原理，构建系统化理论框架。

技能进阶篇：基于ROS编程开发，详解机器人编程基础、客户端库、日志消息、消息录制与回放等，夯实开发技能。

项目实战篇：聚焦Roban机器人、机器人任务挑战赛等典型场景，培养学生复杂工程问题的解决能力。

三、教材特色：紧扣产业需求，创新育人模式

1. 思政融合，铸魂育人

深入挖掘工匠精神、科技报国等思政元素，将职业伦理、社会责任融入技术教学，培养具有家国情怀和创新精神的新时代技术人才。

2. 产教协同，标准对接

联合乐聚机器人、双高院校专家团队，将企业真实项目转化为教学案例，确保教学内容与岗位能力需求“同频共振”。

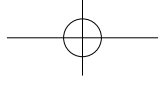
3. 岗课赛证融通，赋能职业发展

深度对接“1+X”工业机器人操作与运维证书标准，拆解ROS技术微技能单元；整合仿人形机器人任务挑战赛等赛事资源，设置阶梯式挑战任务，为学生铺设“学历证书+若干技能证书”的复合型成长通道。

ROS技术正在重塑智能制造的未来，而掌握其核心技术将成为从业者的核心竞争力。希望读者通过系统学习，不仅能够熟练运用ROS开发工具，更能深刻理解技术报国的使命，在智能制造的浪潮中勇担时代重任，为我国建设制造强国贡献青春力量！

职业技能清单及岗位提升路线

岗位等级	岗位名称	岗位职责	知识要求	技能要求	素质要求	章节知识点对应
初级	ROS 系统运维助理	协助工程师进行 ROS 系统的日常维护，包括环境检查、软件更新、数据备份等；处理简单的系统故障和错误信息收集；在指导下参与小型机器人项目的基础搭建工作，如硬件连接与初步配置	了解机器人操作系统的基本概念与架构；熟悉 Ubuntu 操作系统的基本操作命令；掌握 ROS 系统安装与配置的基本流程	能够按照指导完成 ROS 系统的简单安装与设置；会使用基本的命令行工具检查系统状态；能协助进行简单的硬件设备连接与驱动安装	具备良好的学习能力和团队协作精神；有较强的责任心和耐心，能认真完成基础的运维任务；对机器人技术有一定的兴趣和热情	对应教材中“模块一掌握 ROS 核心基础”章节，如 ROS 系统架构与生态、Ubuntu 操作系统安装、ROS 环境部署等知识点
	机器人测试技术员	按照测试规范对机器人进行功能性测试，包括硬件功能测试（如传感器精度、电机性能等）和软件功能测试（如 ROS 程序的基本功能、通信稳定性等）；记录测试数据并反馈测试结果，协助分析和定位测试中发现问题；参与测试设备的维护与校准工作	熟悉机器人硬件的基本组成和工作原理；了解 ROS 系统的基本操作和常见功能；掌握基本的测试理论和方法	能够熟练操作测试设备对机器人进行测试；会使用数据记录工具准确记录测试数据；能在指导下对简单测试问题进行初步排查	工作认真细致，有较强的观察力和数据记录能力；具备一定的学习能力，能快速掌握新的测试技术和方法；有良好的团队协作精神	对应“模块一掌握 ROS 核心基础”中关于 ROS 系统基本操作和“模块三完成 Roban 机器人运动控制实战”中机器人硬件相关知识的初步了解部分，以及测试相关基础知识
中级	ROS 应用开发工程师	负责开发基于 ROS 的机器人应用程序，实现机器人的基本运动控制、传感器数据处理与简单的任务逻辑；参与机器人项目的功能模块设计与编码实现；对开发过程中出现的问题进行调试与优化；与团队成员协作完成项目的集成与测试工作	深入理解 ROS 的通信机制（话题、服务、动作库等）；掌握 C++ 或 Python 语言在 ROS 开发中的应用；熟悉常见的机器人传感器原理与类型；了解机器人运动控制的基本算法	熟练使用 C++ 或 Python 编写 ROS 节点程序实现话题通信、服务通信；能够自定义消息和服务类型；会利用 ROS 工具进行调试和性能分析；能独立完成简单机器人应用的开发与部署	具备较强的问题解决能力和逻辑思维能力；有良好的编程习惯和代码规范意识；能够在团队中有效沟通与协作，共同推进项目进展	涵盖“模块二实现 ROS 运动控制开发”和“模块四集成 ROS 客户端库与算法”章节，如 ROS 通信机制、自定义消息类型、客户端库应用等知识点



(续表)

中级	机器人数据分析师	收集和整理机器人运行过程中的各类数据,包括传感器数据、运动轨迹数据、任务执行数据等;运用数据分析工具和方法对数据进行处理和分析,提取有价值的信息,如机器人性能趋势、故障预测等;为机器人的优化和改进提供数据支持和建议;与开发团队和运维团队协作,共同解决数据相关问题	掌握数据分析的基本理论和方法,如统计学、数据挖掘等;熟悉常见的数据分析工具,如 Python 的数据分析库 (numpy、pandas 等);了解机器人数据的特点和来源	能够使用数据分析工具对机器人数据进行清洗、分析和可视化;会编写简单的数据处理脚本;能根据数据分析结果提出合理的改进建议	具备较强的数据分析能力和逻辑思维能力;有良好的沟通能力,能与不同团队有效协作;对数据有敏锐的洞察力,善于发现数据中的规律和问题	与“模块六优化 ROS 数据记录与回放”相关,涉及数据的收集、处理和分析方法,以及部分“模块四集成 ROS 客户端库与算法”中关于数据处理相关的知识
	机器人系统架构师	主导机器人系统的整体架构设计,根据项目需求规划 ROS 系统的功能模块与技术选型;带领团队进行复杂机器人项目的开发与实施,确保系统的高性能、高可靠性和可扩展性;负责解决项目中的关键技术难题,对新技术进行研究与应用;与客户或其他部门进行技术沟通与协调,确定项目需求与技术方案	精通 ROS 系统的高级特性与应用场景;掌握机器人领域的前沿技术与发展趋势;熟悉多种机器人硬件平台的架构与性能特点;具备系统工程的相关知识 with 经验	能够设计和优化复杂的 ROS 系统架构;熟练运用多种编程技术和工具进行高效的机器人系统开发;对机器人系统的性能瓶颈有敏锐的洞察力,并能提出有效的解决方案;有能力领导团队完成大型机器人项目的开发与交付	拥有卓越的技术领导力和团队管理能力;具备创新思维和战略眼光,能引领团队技术发展方向;具有良好的沟通协调能力和项目管理能力,确保项目顺利推进	综合运用教材各模块的高级知识点,如 ROS 系统的优化与扩展、复杂机器人应用的架构设计、多传感器融合与智能算法应用等,并需不断关注行业前沿知识与技术更新
高级	机器人项目技术经理	负责机器人项目的整体技术规划和管理,制定项目技术路线和开发计划;组织和协调团队成员进行项目开发,合理分配资源,确保项目按时交付;对项目中的技术难题进行决策和指导,把控项目技术风险;与客户、合作伙伴等进行技术沟通和商务洽谈,推动项目的顺利进行	精通机器人技术的各个方面,包括硬件、软件、算法等;熟悉项目管理的流程和方法;了解行业动态和市场需求	具备丰富的机器人项目开发经验,能够独立完成复杂项目的技术架构设计;熟练掌握项目管理工具和技术,有效管理项目进度、质量和成本;有出色的沟通和谈判能力,能处理好项目中的各种技术和商务关系	具有卓越的领导能力和团队管理能力,能够激励和带领团队完成项目目标;具备敏锐的市场洞察力和战略眼光,能把握项目的技术方向和市场机会;有很强的抗压能力和问题解决能力,应对项目中的各种挑战	综合运用教材全部章节的知识,并结合项目管理、市场分析等外部知识,需要对机器人技术有全面深入的理解和丰富的实践经验

Contents 目录



模块一 掌握 ROS 核心基础

01

003-030

项目一

理解 ROS 系统架构与生态

- 1.1 认识 ROS 核心概念与设计思想
- 1.2 安装并配置 Ubuntu 操作系统
- 1.3 完成 ROS 环境部署与验证
- 1.4 总结 ROS 基础能力要点

02

031-064

项目二

解析 ROS 系统架构与通信机制

- 2.1 分析 ROS 软件架构与模块交互
- 2.2 掌握 ROS 文件系统组织规范
- 2.3 实践 ROS 节点通信
- 2.4 总结 ROS 架构设计原则



03

067-088

项目三

配置 ROS 开发工具链

- 3.1 搭建 RoboWare Studio 集成开发环境
- 3.2 使用 Git 进行代码版本管理
- 3.3 运行 Gazebo 仿真环境与模型调试
- 3.4 实现 Rviz 可视化数据监控
- 3.5 应用 rqt 工具优化调试流程
- 3.6 总结工具链协同使用方法

模块二 实现 ROS 运动控制开发

04

089-121

项目四

开发 ROS 基础通信功能

- 4.1 对比 ROS 与传统编程范式差异
- 4.2 使用 C++ 实现话题通信
- 4.3 使用 Python 实现服务通信
- 4.4 定义并编译自定义消息
- 4.5 定义并编译服务类型
- 4.6 编写服务端与客户端程序
- 4.7 设计多节点协同启动文件
- 4.8 总结通信开发核心技术



05

123-146

项目五

集成 ROS 客户端库与算法

- 5.1 调用 C++ 接口实现运动控制逻辑
- 5.2 调用 Python 接口实现传感器数据处理
- 5.3 实现日志消息生成与动态管理
- 5.4 总结客户端库应用场景

06

147-166

项目六

优化 ROS 数据记录与回放

- 6.1 录制 ROS 数据包并分析内容
- 6.2 回放数据包复现运动过程
- 6.3 集成数据包到启动文件实现自动化
- 6.4 总结数据管理最佳实践
- 6.5 本章小结



模块三 完成 Roban 机器人运动控制实战

07

167-178

项目七

部署 Roban 机器人开发环境

- 7.1 认识 Roban 硬件结构与功能特性
- 7.2 配置 Roban 机器人通信接口
- 7.3 通过 PC 端软件操控 Roban 基础运动
- 7.4 总结 Roban 开发环境搭建要点

08

179-202

项目八

实现仿人形机器人建图导航任务

- 8.1 设计 ORB-SLAM2 多传感器融合建图方案
- 8.2 开发基于 ROS 的路径规划与避障算法
- 8.3 调试并验证机器人自主导航性能
- 8.4 总结运动控制任务实现流程

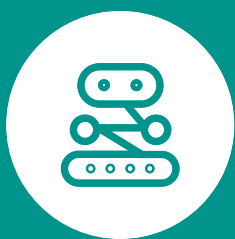
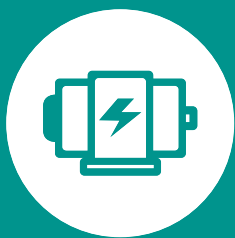
09

203-227

项目九

挑战机器人运动控制综合任务

- 9.1 定义多模态运动控制任务目标
- 9.2 实现步态生成与动态平衡算法
- 9.3 优化任务代码并完成性能测试
- 9.4 总结机器人运动控制技术趋势



01

模块一 掌握 ROS 核心基础

项目一

理解 ROS 系统 架构与生态



● 技术先锋

中国航天科技集团的特级技师徐立平，从事导弹固体燃料发动机的火药微整形工作，其操作精度误差不超过 0.2 毫米，堪称“雕刻火药的大国工匠”。他在极度危险的岗位上，凭借着对技术的执着和对国家航天事业的热爱，以精益求精的态度完成每一次操作。

ROS 系统的学习恰似徐立平的工作，从基础概念到开发环境搭建，每一步都需精准无误。在了解 ROS 起源、功能特点以及发展历程时，如同徐立平钻研火药微整形技术原理，我们要深入探究其背后的逻辑。搭建 Ubuntu 操作系统和 ROS 环境，配置 VMware 虚拟机等操作，就像徐立平的精细操作，任何一个小的失误都可能影响后续学习与实践。我们应学习徐立平严谨认真、追求极致的精神，确保对 ROS 系统基础的扎实掌握，为后续学习筑牢根基。

机器人操作系统 (Robot Operating System, ROS) 是一个应用于机器人系统的通用软件框架。ROS 诞生以来，受到了工业界、学术界和科研机构的欢迎，如今已广泛应用于工业机器人、移动机器人、无人车、无人机等多种类的机器人上，取得了良好的应用效果。

本章从介绍 ROS 的起源开始，简要介绍 ROS 的发展历程、功能与特点，以帮助读者对 ROS 有一个初步的了解。同时，详细介绍 Ubuntu (乌班图) 操作系统和 ROS 的安装与配置方法等，以帮助读者搭建 ROS 系统的开发环境，为后续基于 ROS 的编程和应用程序开发做好准备。

1.1

认识 ROS 核心概念与设计思想

1.1.1

ROS 的起源



ROS 简介及历史起源

1886 年，法国作家利尔·亚当在他的小说《未来的夏娃》中，将外表像人的机器起名为“Android（安卓）”；1920 年，捷克剧作家卡雷尔·凯培克在他的幻想情景剧《罗素姆万能机器人》中，把捷克语的“Robota（原意为劳役、苦工）”写成了“Robot”，后来逐渐演化为机器人的专用名词。早期文学和戏剧作品中的机器人如图 1.1 所示。



图 1.1 早期文学和戏剧作品中的机器人

早期文学和戏剧作品中的机器人形象更多来自于文学家、剧作家的想象。世界上第一台实用型工业机器人是美国 AMF 公司在 1962 年推出的 UNIMATE 机器人，如图 1.2 所示。



图 1.2 世界上第一台实用型工业机器人 UNIMATE

随着机器人技术的快速发展，机器人平台与硬件设备越来越丰富，也越来越复杂。不同厂商的机器人产品可能拥有不同的软、硬件系统及其接口标准。丰富多样的现代机器人如图 1.3 所示。

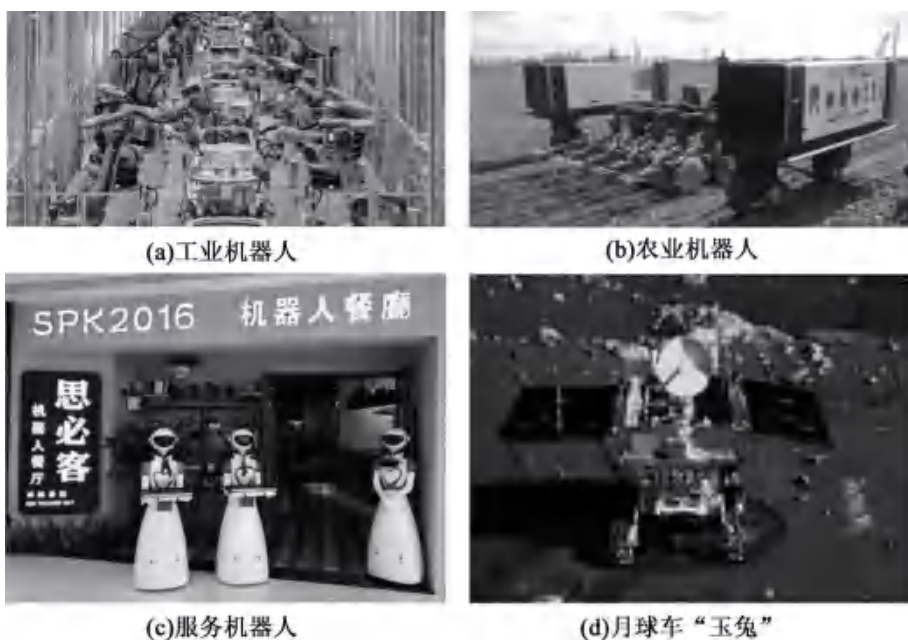


图 1.3 丰富多样的现代机器人

对于机器人开发人员来说，如果从头构建机器人，需要解决传感器、执行器等硬件设备的驱动问题，以及底层运动的控制问题，这势必会消耗开发人员太多的时间去做一些重复性的工作，如传感器的数据采集、机器人视觉的图像处理、伺服驱动控制等，从而使得开发人员没有精力和时间去开发更有意义和价值的高级功能、人工智能等方面的功能。因此，机器人领域软件代码的复用、模块化编程的需求越来越强烈。

为此，全球各地的开发者、研究机构、科技公司等投入了大量人力和物力到机器人通用软件平台的研发工作中，也产生了许多机器人软件平台，如 Player、YARP、OROCOS、Carmen、Orca、MOOS 和 Microsoft Robotics Studio 等。其中，非常优秀的软件平台之一就是机器人操作

系统（Robot Operating System，ROS）。

ROS 的原型系统起源于 2007 年美国斯坦福大学人工智能实验室与 Willow Garage 公司合作的 Personal Robotics Program（机器人管家项目）。随着该团队研发的二代机器人（PR2）在 ROS 框架的基础上实现了叠衣服、做早饭等不可思议的功能，如图 1.4 所示，ROS 系统也得到越来越多的关注。

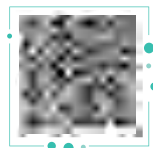


图 1.4 PR2 机器人的典型应用场景

2009 年初，Willow Garage 公司推出了 ROS0.4 测试版，标志着 ROS 系统框架已经具备雏形；2010 年，Willow Garage 公司以开放源代码的形式发布了 ROS1.0 版本，很快就在机器人研究领域掀起了学习与应用 ROS 的热潮；2012 年，Willow Garage 公司将 ROS 交由一家新成立的非营利性机构—开源机器人基金会（Open Source Robotics Foundation，OSRF）来维护，旨在通过接受个人、公司和政府的赞助，独立地推动 ROS 以开源的方式发展；从 2013 年开始，ROS 系统得到了 Google 公司的支持，对开发人员而言，ROS 的开放性和易用性使其获得了广泛应用，而 Google 公司持续的支持是 ROS 存活的关键。

1.1.2

ROS 的功能与特点



ROS 五大特点

ROS 网站（<https://www.ros.org>）对 ROS 的定义如下。

机器人操作系统（ROS）是一组帮助软件开发者构建机器人应用程序的软件库和工具集。从驱动程序到最先进的算法，以及功能强大的开发工具，ROS 为软件开发者提供了机器人项目所需的一切，而且都是开源的。

ROS 维基百科中文网站（<https://wiki.ros.org/cn>）对 ROS 的功能介绍如下。

ROS 提供一系列程序库和工具，帮助软件开发者创建机器人应用软件。它提供了硬件抽象、设备驱动、函数库、可视化工具、消息传递和功能包管理等诸多功能。ROS 遵循 BSD 开源许

可协议（Berkeler Software Distribution License）。

ROS 不仅能够实现对机器人运动位置控制、姿态轨迹规划、操作顺序管理、人机交互及多机协同等功能，还能够支持机器人软件与系统的仿真、开发、测试与验证等环节，其主要特点如下。

1. 分布式、点对点 (P2P) 设计

ROS 将应用程序的每个工作进程都看作一个节点（Node），使用节点管理器（Node Master）进行统一管理，并提供了一套消息传递机制。这些节点可以在一台计算机上运行，也可以在网络上的多台计算机上运行，从而实现了分布式计算。这种点对点的设计可以分散由定位、导航、视觉识别、语音识别等功能带来的实时计算压力，适应多机器人的协同工作。

2. 支持多种语言

ROS 编程语言，目前已经支持 C++、Python、Lisp、Java、Octave 等多种现代编程语言。为了支持多语言编程，ROS 使用了一种独立于编程语言的接口定义语言（Interface Definition Language, IDL）来描述模块之间的消息接口，并且实现了多种编程语言对 IDL 的封装，从而使得开发者可以同时使用多种编程语言来完成不同模块的开发。

3. 精简与集成

ROS 框架具有的模块化特点，使得每个功能模块代码可以单独编译，并且使用统一的消息接口让模块的移植和复用更加便捷。同时，ROS 开源社区中集成了大量已有开源项目中的代码。例如，从 Player 项目中借鉴了驱动、运动控制和仿真方面的代码，从 OpenCV 中借鉴了视觉算法方面的代码，从 OpenRAVE 中借鉴了规划算法方面的代码。开发者可以利用这些资源实现机器人应用的快速开发。

4. 丰富的组件化工具包

ROS 采用组件化的方式将已有的工具和软件进行集成，比如 ROS 中的三维可视化平台 Rviz，它是 ROS 自带的一个图形化工具，可以方便地对 ROS 的应用程序进行图形化操作。再比如 ROS 常用的物理仿真平台 Gazebo，在该仿真平台下，不仅可以创建一个虚拟的机器人仿真环境，还可以在仿真环境中设置一些必要的参数。

5. 开源且免费

ROS 的源代码全部公开发布，从而极大地提升了 ROS 框架各层次错误更正的效率。同时，ROS 遵循 BSD 开源许可协议，给用户较大的自由，允许个人修改和发布新的应用，甚至可以进行商业化开发和销售。这就使得 ROS 拥有强大的生命力。在短短的几年内，ROS 功能包的数量呈指数级增长，从而大大加速了机器人应用的开发。

1.1.3

ROS 的发展历程与趋势

ROS 1.0 版本发布于 2010 年，是基于 PR2 机器人开发的一系列基础功能包。类似于 Linux 的发行版，例如 Ubuntu，ROS 发行版的目的是让开发人员在一个相对稳定的代码库上工作。

随着 ROS 发行版本的不断迭代，常用的 ROS 版本与首选的 Ubuntu 版本见表 1.1，目前已经发布到了 Noetic 版本。本书选择 ROS Kinetic + Ubuntu 16.04 + Python 2.7 版本的组合，一方面是因为该组合在 Wiki 网站上拥有众多的学习资源，方便读者后续的自学与提高；另一方面是为了与第 8 章介绍的 Roban（中型仿人形机器人）相配合，完成 Roban 机器人挑战赛的任务。

表 1.1 常用的 ROS 版本与首选的 Ubuntu 版本

发布时间	ROS 版本	首选的 Ubuntu 版本	停止支持时间
2010 年 3 月	Box Turtle		
2010 年 8 月	C Turtle		
2011 年 3 月	Diamondback		
2011 年 8 月	Electric Emys		
2012 年 4 月	Fuerte Turtle		
2012 年 12 月	Groovy Galapagos		2014 年 7 月
2013 年 9 月	Hydro Medusa		2015 年 5 月
2014 年 7 月	Indigo Igloo	Ubuntu 14 .04	2019 年 4 月
2015 年 5 月	Jade Turtle	Ubuntu 15 .04	2017 年 5 月
2016 年 5 月	Kinetic Kame（本书安装的版本）	Ubuntu 16 .04	2021 年 4 月
2017 年 5 月	Lunar Loggerhead	Ubuntu 17 .04	2019 年 5 月
2018 年 5 月	Melodic Morenia	Ubuntu 18 .04	2023 年 5 月
2020 年 5 月	Noetic Ninjemys(Wiki 推荐的版本)	Ubuntu 20 .04	2025 年 5 月

作为一个开放源代码的软件系统，ROS 构建了一个能够整合不同研究成果，实现算法发布、代码重用的机器人通用软件平台，解决了机器人技术研发中大量的共性问题，但依然存在很多缺陷和不足。如 ROS 中没有构建多机器人系统的标准方法，ROS 无法在 Windows、MacOS 等操作系统上应用，或者说功能是受限的，ROS 缺少实时性方面的设计，ROS 系统整体运行效率较低。总体来说，ROS 的稳定性欠佳使得机器人技术研发从原型样机到最终产品的产品化过程非常艰难。

为此，在 ROSCon 2014 年会上，OSRF 正式发布了新一代的 ROS 设计架构（Next-generation ROS:BuildingonDDS），即 ROS2，并在 2015 年 8 月发布了 ROS2 的 Alpha 版本，2016 年 12 月发布了 Beta 版。随后，OSRF 于 2017 年 12 月发布了 ROS2 的第一个发行版本 Ardent Apalone，从

而开创了下一代 ROS 开发的新纪元。

相比之前的 ROS 系统，ROS 2 的改进主要采用了 DDS（数据分发服务）这个工业级的中间件来负责可靠通信、通信节点动态发现，并采用共享内存的方式来提高通信的效率，从而让 ROS 2 更符合工业级的运行标准。

ROS 2 是 ROS 系统的功能扩展和性能优化，其设计目标主要体现在以下几个方面。

- （1）支持多机器人系统，包括不可靠的网络。
- （2）消除原型样机和最终产品之间的鸿沟。
- （3）可以运行在小型嵌入式平台上。
- （4）支持实时控制。
- （5）提供跨系统平台支持。

2022 年 5 月，OSRF 发布了 ROS 2 的第一款长期支持版本——Humble LTS（Long Term Sport），支持期为 5 年。

从总体来看，目前 ROS 2 还处于发展的初级阶段，而 ROS 开源社区已经积累了丰富的资源，出版了大量的 ROS 书籍，也启动了许多教育培训计划。因此，目前机器人技术的研究、开发与教学仍然建议以 ROS 为主。

1.2

安装并配置 Ubuntu 操作系统

ROS 并不是一个传统意义上的操作系统，无法像 Windows、Mac OS、Linux 系统一样直接运行在计算机硬件上，而是需要依托于 Linux 操作系统。Ubuntu 是 Linux 操作系统的一个发行版本，由英国公司 Canonical（www.canonical.com）负责运行和维护。Ubuntu（www.ubuntu.com）可以被免费下载和使用，也是目前对 ROS 支持最好的操作系统，本书采用 Ubuntu 16.04 + ROS Kinetic 版本的组合。

1.2.1

安装 VMware 虚拟机

VMware 是一个虚拟化软件，它允许一个未经修改的操作系统（包含全部已安装软件）运行在一个被称为虚拟机的特殊环境中。虚拟机运行在当前操作系统之上，是由虚拟化软件通过拦截对某些硬件和功能的访问实现的。物理实体计算机称为宿主机，虚拟机称为客户机，客户机可以运行在宿主机之上。

可以在一台运行 Windows、Linux 或者 Mac OS 操作系统的计算机上安装 VMware。本书将选择 VMware Workstation Pro 版本，安装在一台装有 Windows 系统的计算机上。VMware 的安装很简单，依次点击“下一步”选项即可，如图 1.5 所示。

需要说明的是，需要选择默认的安装路径，即 C:\Program Files（x86）目录下。



图 1.5 安装 VMware 虚拟机

1.2.2

在虚拟机上安装 Ubuntu 操作系统



Ubuntu 安装

在 VMware 上安装 Ubuntu，首先要创建一个新的虚拟机。

第 1 步：新建一个虚拟机。

在计算机上打开 VMware 虚拟机，如图 1.6 所示，选择“典型（推荐）（T）”，然后点击“下一步”按钮。



图 1.6 新建一个 VMware 虚拟机

在“新建虚拟机向导”界面中，选择“稍后安装操作系统（S）”选项，如图 1.7 所示，然后点击“下一步”按钮。



图 1.7 新建虚拟机向导

选择客户机操作系统为“Linux（L）”，版本（V）为“Ubuntu64 位”，如图 1.8 所示，然后点击“下一步”按钮。



图 1.8 选择操作系统和版本

第 2 步：为客户机操作系统命名。

添加完 VMware 虚拟机之后，需要为将要创建的客户机操作系统命名（虚拟机名称），并选择安装位置，如图 1.9 所示。设置完成后，点击“下一步”按钮。



图 1.9 为客户操作系统操作系统命名并选择安装位置

第 3 步：为客户操作系统分配磁盘空间。

设置虚拟机磁盘大小，一般设置为 40.0 GB，将虚拟磁盘存储为单个文件，如图 1.10 所示，然后点击“下一步”按钮。

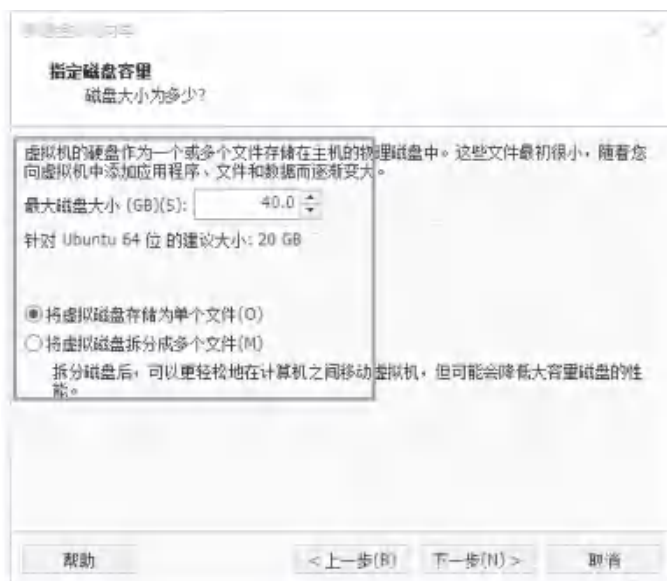


图 1.10 为客户机分配磁盘空间

查看虚拟机的设置情况，如图 1.11 所示，确认无误后点击“完成”按钮。



图 1.11 完成虚拟机的创建

第 4 步：设置镜像。

创建完虚拟机后，还需要设置镜像，点击“编辑虚拟机设置”按钮，如图 1.12 所示。



图 1.12 编辑虚拟机设置

在 CD / DVD（SATA）中选择“使用 ISO 映像文件（M）”，加载 ISO 镜像文件，如图 1.13 所示。

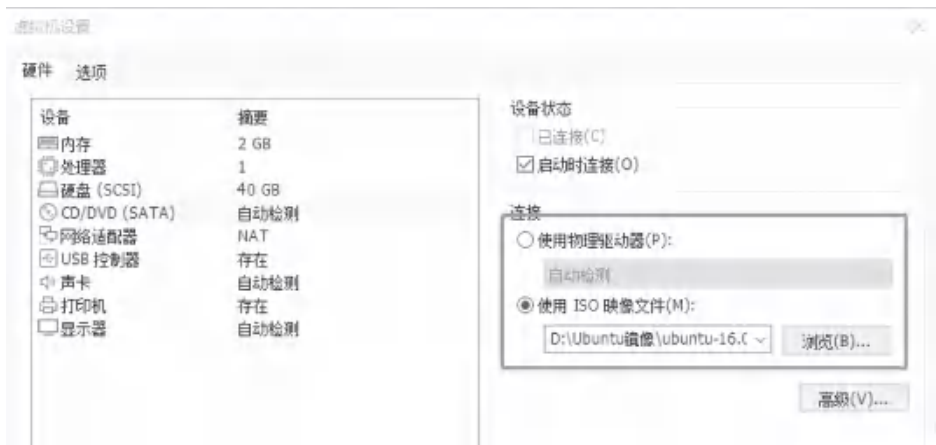


图 1.13 加载 ISO 镜像文件

第 5 步：开启虚拟机，安装 Ubuntu 操作系统。

完成虚拟机设置和 ISO 镜像文件的设置后，开启虚拟机，进入如图 1.14 所示的安装界面。在左侧栏中选择语言“中文（简体）”或“English”；在右侧栏中选择“安装 Ubuntu”，开始安装 Ubuntu 操作系统。



图 1.14 安装 Ubuntu 操作系统

在如图 1.15 所示的界面中，如果是在虚拟机上安装 Ubuntu 系统，则可以忽略这一步；如果是在一台装有显卡的计算机上直接安装 Ubuntu 系统，如装有 NVIDIA 或 ATI Radeon 显卡的计算机，则需要选择“安装 Ubuntu 时下载更新”这个选项，它可以搜索一个合适的显卡驱动程序并在 Ubuntu 安装过程中安装它，否则可能需要手动安装。但是，这一过程并不能确保可以获得适用于自己的显卡驱动程序。

配置完成后，点击“继续”按钮。



图 1.15 安装 Ubuntu 时下载更新

在如图 1.16 所示的界面中, 将对硬盘进行分区, 并在其上安装 Ubuntu, 这个步骤非常重要, 用户必须谨慎地选择分区方式。



图 1.16 清除整个磁盘并安装 Ubuntu

第一个选项“清除整个磁盘并安装 Ubuntu”，将擦除硬盘上的所有硬盘分区并安装 Ubuntu。如果是在虚拟机中安装 Ubuntu，这个选项很合适；而如果是在实体计算机上直接安装 Ubuntu，则应选择“其他选项”。“其他选项”允许用户格式化特定的硬盘分区并在其上安装 Ubuntu。如果在虚拟机中安装 Ubuntu，则不必关心这个问题，因为虚拟机中只有一个硬盘。而如果是在实体计算机上安装 Ubuntu，就必须在安装操作系统之前找到一个安装 Ubuntu 的分区。

安装 Ubuntu 时，通常需要创建两个分区：一个是根分区，另一个是交换分区。Ubuntu 操作系统安装在根分区中。文件系统的格式是 Ext4journaling, 必须将根分区的挂载点设置为“/”。交换分区是一种内存接近最大使用率时用于存储非活动界面的特殊分区，类似于 Windows 中虚拟内存的概念。如果计算机的内存足够大，比如大于 4.0 GB, 则可以不用设置交换分区，否则需要设置一个交换分区。用户可以分配 1 GB 或 2 GB 内存给交换分区。

当两个分区创建完成后，在如图 1.16 所示的界面中点击“现在安装”按钮，将弹出“将改动写入磁盘吗？”界面，如图 1.17 所示。点击“继续”按钮，把 Ubuntu 安装到所选分区中。在安装过程中，可以设置时区、键盘布局、用户名和密码等。



图 1.17 将改动写入磁盘

设置时区时，在弹出的“您在什么地方？”界面中，用户可以点击所在国家地图来进行。当点击地图时，将会显示国家的名称，即选择了 Ubuntu 系统的时区。点击“继续”按钮，设置键盘布局。

设置时区后，下一步是设置键盘布局，如图 1.18 所示。用户可以选择使用默认的键盘布局 [English (US)]，也可以选择中文的键盘布局（汉语）。点击“继续”按钮，设置计算机名称、用户名和密码。

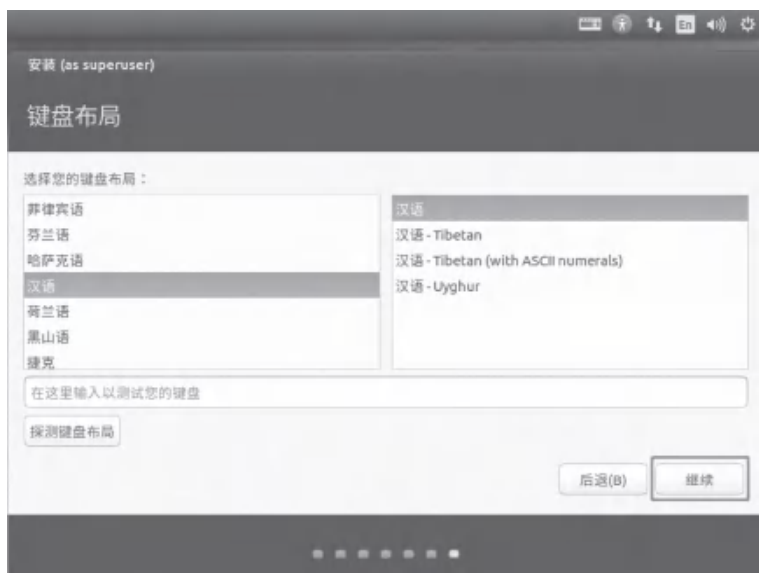


图 1.18 选择键盘布局

在设置用户名和登录密码界面中，如图 1.19 所示，如果不想使用用户名和密码登录，可启用自动登录功能，今后将无须输入用户名和密码而直接登录 Ubuntu 桌面。点击“继续”按钮，完成后续的安装。



图 1.19 设置用户名和密码

在输入完用户名和登录密码之后，安装 Ubuntu 系统的设置过程就已经全部完成。系统开始安装，用户需要等待一段时间。Ubuntu 系统安装完成后，需要重启虚拟机。

重启之后，将在 VMware 界面中看到 Ubuntu 主界面，如图 1.20 所示，说明已经在虚拟机上成功地安装了 Ubuntu 操作系统。

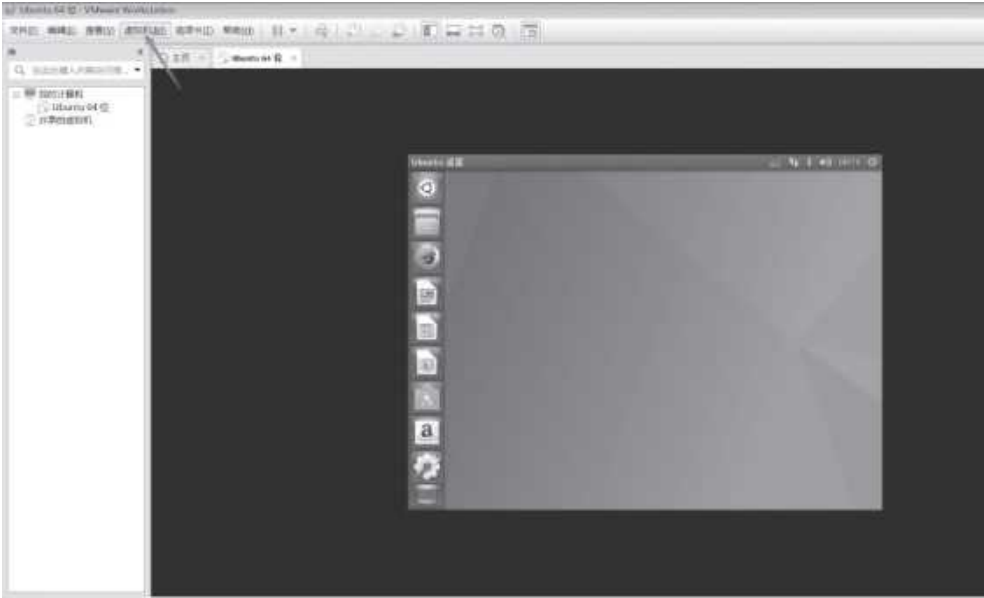


图 1.20 在 VMware 虚拟机中成功安装 Ubuntu 操作系统

第 6 步：安装 VMwareTools。

VMware Tools 是虚拟机上一个很好用的工具，很多地方都需要用到，是一个必装的工具。首先取消 ISO 镜像文件设置，如图 1.21 所示。



图 1.21 取消 ISO 镜像文件设置

选择虚拟机菜单栏→安装 VMwareTools。下载好后在 Ubuntu 系统中弹出的 VMware Tools 窗口中找到 VMwareTools - 10 .3 .10 - 13959562 .tar.gz 文件，复制到桌面上，如图 1.22 所示。

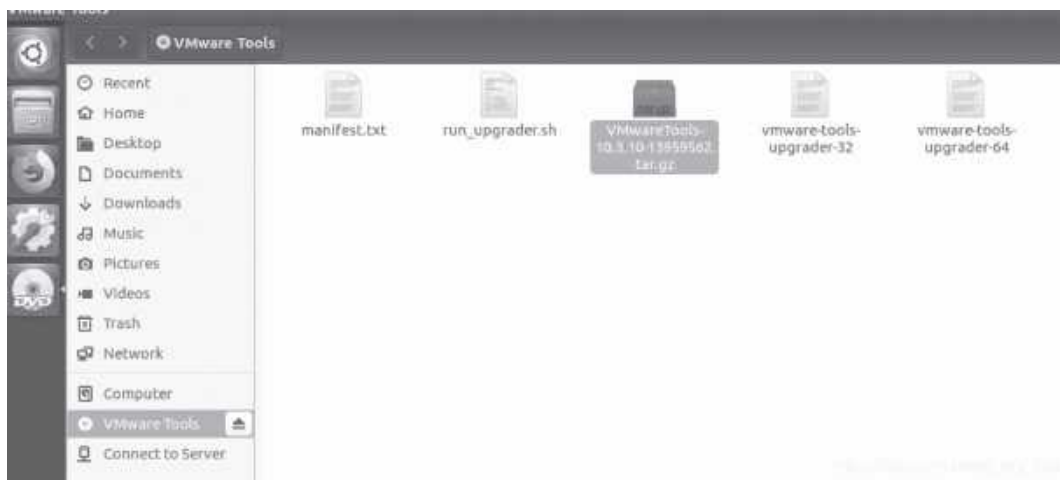


图 1.22 复制 VMware Tools 的安装包

打开一个终端，进入到桌面目录下，输入以下命令开始解压缩文件，如图 1.23、图 1.24 所示。

```
hqb@hqb-virtual-machine: ~/Desktop
hqb@hqb-virtual-machine:~$ ls
Desktop  Downloads  examples.desktop  Pictures  Templates
Documents  esp8266  Music  Public  Videos
hqb@hqb-virtual-machine:~$ cd Desktop/
hqb@hqb-virtual-machine:~/Desktop$
```

图 1.23 进入桌面文件夹

```
hqb@hqb-virtual-machine:~/Desktop$ tar xzf VMwareTools-10.3.10-13959562.tar.gz
hqb@hqb-virtual-machine:~/Desktop$ ls
VMwareTools-10.3.10-13959562.tar.gz  vmware-tools-distrib
hqb@hqb-virtual-machine:~/Desktop$
```

图 1.24 解压缩文件

进入到 vmware-tools-distrib 文件夹下，运行 vmware-install.pl，如图 1.25 所示。

```
hqb@hqb-virtual-machine:~/Desktop$ cd vmware-tools-distrib/
hqb@hqb-virtual-machine:~/Desktop/vmware-tools-distrib$ ls
bin  bin  doc  etc  FILES  INSTALL  installer  lib  vgauth  vmware-install.pl
hqb@hqb-virtual-machine:~/Desktop/vmware-tools-distrib$ sudo ./vmware-install.pl

[sudo] password for hqb:
open-vm-tools packages are available from the OS vendor and VMware recommends
using open-vm-tools packages. See http://kb.vmware.com/kb/2073803 for more
information.
Do you still want to proceed with this installation? [yes]
```

图 1.25 运行 vmware – install.pl

之后一直按“回车”键全默认即可，安装完成后需要重新设置 ISO 镜像文件。

第 7 步：设置共享文件夹。

如何实现 Ubuntu 系统和 Windows 系统的文件共享？VMware 虚拟机中自带了共享文件夹的功能。

设置 Ubuntu 系统的共享文件夹，操作如下。

选择 Ubuntu 虚拟机，在菜单栏中，单击虚拟机设置→选项，开启共享文件夹功能，如图 1.26 所示。

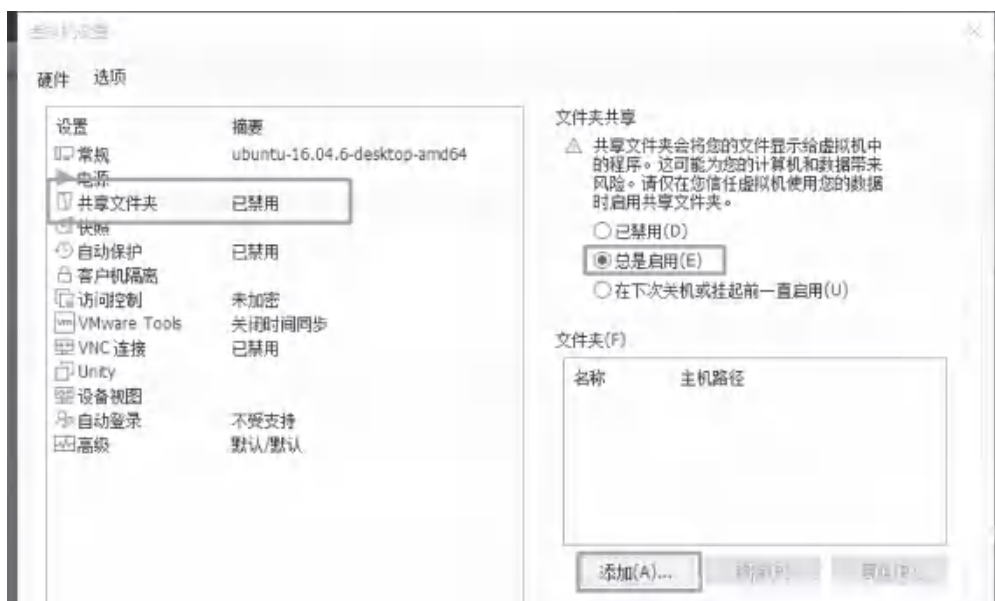


图 1.26 开启共享文件夹功能

添加共享文件夹，如图 1.27 所示。



图 1.27 添加共享文件夹

设置共享文件夹路径，如图 1.28 所示。



图 1.28 设置共享文件夹路径

启用共享文件夹，如图 1.29 所示。



图 1.29 启用共享文件夹

共享文件夹设置完成，如图 1.30 所示。

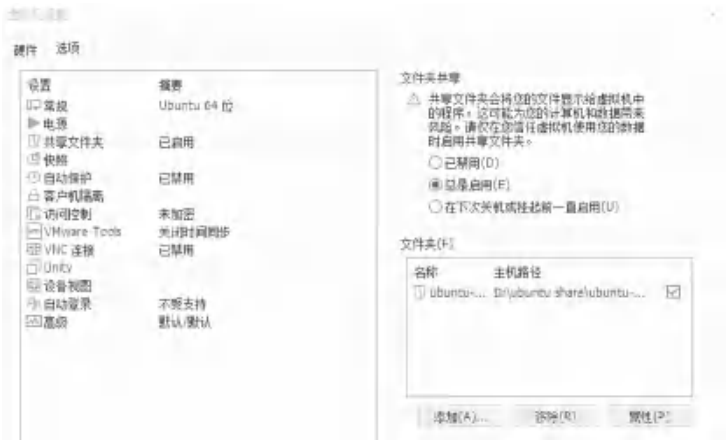


图 1.30 共享文件夹设置完成

1.2.3

在计算机上直接安装 Ubuntu 操作系统

在一台计算机上直接安装 Ubuntu 操作系统通常有两种方式。第一种方式很直接，首先将下载的 DVD 镜像刻录到一个 DVD 光盘上，然后从 DVD 启动安装。另一种方式是用 USB 驱动器安装，它比前一种方式更容易，也更快。

一款名为 UNetbootin 的工具可以将 DVD 镜像复制到 USB 驱动器。用户可以通过这款工具浏览 DVD 镜像，点击“确定”按钮开始复制过程（UNetbootin 设置如图 1.31 所示）。

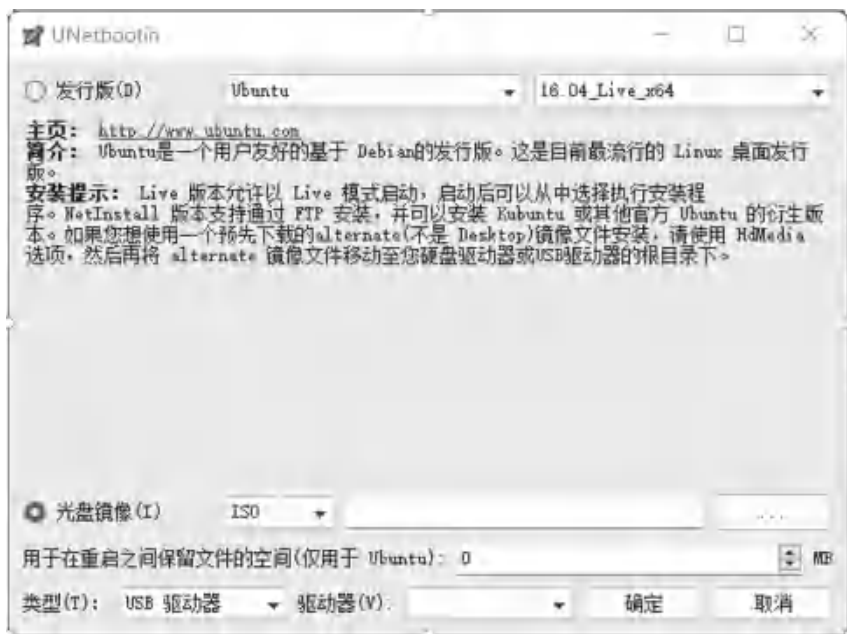


图 1.31 UNetbootin 设置

用户可以选择 Ubuntu 系统并浏览其 DVD 镜像。在选择 DVD 镜像之后，选择 USB 驱动类型，接下来选择驱动器盘符字母，最后点击“确定”按钮。把 DVD 镜像复制到 USB 驱动器上需要花费一段时间。

当它完成后，重新启动计算机，将首选启动设备设置为 USB 驱动器。此时系统会从 USB 驱动器启动安装，读者也可以参照前述的“在 VMware 虚拟机中安装 Ubuntu 操作系统”的过程进行安装。

1.3

完成 ROS 环境部署与验证

1.3.1

ROS 版本的选择



ROS 的安装

上文已经说明本书采用 Ubuntu16.04 + ROS Kinetic 版本的组合。

ROS 中有很多函数库和工具，官网提供了四种默认的安装方式：桌面完整版（Desktop - Full）安装、桌面版（Desktop）安装、基础版（ROS - Base）安装和单独功能包安（Individual Package）安装。用户根据自己的需要选择其中一种安装方式即可。

（1）桌面完整版（Desktop - Full）：最为推荐的一种安装版本，除了包含 ROS 的基础功能（核心功能包、构建平台和通信机制）外，还包含丰富的机器人通用函数库和功能包，如自主导航、2D/3D 的感知功能、机器人地图建模等，以及 Gazebo 仿真工具、Rviz（the Robot Visualization Tool）可视化平台、rqt 工具箱等。

（2）桌面版（Desktop）：可以理解为桌面完整版的精简版，仅包含 ROS 的基础功能、机器人通用函数库、Rviz 可视化平台和 rqt 工具箱。

（3）基础版（ROS - Base）：仅保留了没有任何 GUI（Graphical User Interface）的基础功能（核心功能、构建工具和通信机制）。基础版是 ROS 需求的“最小系统”，非常适合直接安装在对性能和空间要求较高的控制器上，为嵌入式系统使用 ROS 提供了可能。

（4）单独安装功能包（Individual Package）：在运行 ROS 时，若缺少某些功能包（Package）依赖，就可以单独安装某个指定的功能包。

1.3.2

安装 ROS

1. 检查 Ubuntu 的初始环境

在正式安装 ROS 系统之前，需要先检查 Ubuntu 的初始环境，以保证 ROS 能够正确地安装成功。

打开 Ubuntu 的“System Settings”，依次选择“Software & Updates”→“Ubuntu Software”，勾选关键字“main”“universe”“restricted”和“multiverse”11 项，如图 1.32 所示，设置 Ubuntu 的软件源。

在 Ubuntu 中有以下四种软件源。

- (1) main: Ubuntu 官方支持的免费、开源的软件源。
- (2) universe: 社区维护的免费、开源的软件源。
- (3) retriected: 私有的设备驱动程序软件源。
- (4) multiverse: 该软件源中的软件受到版权和法律的保护。

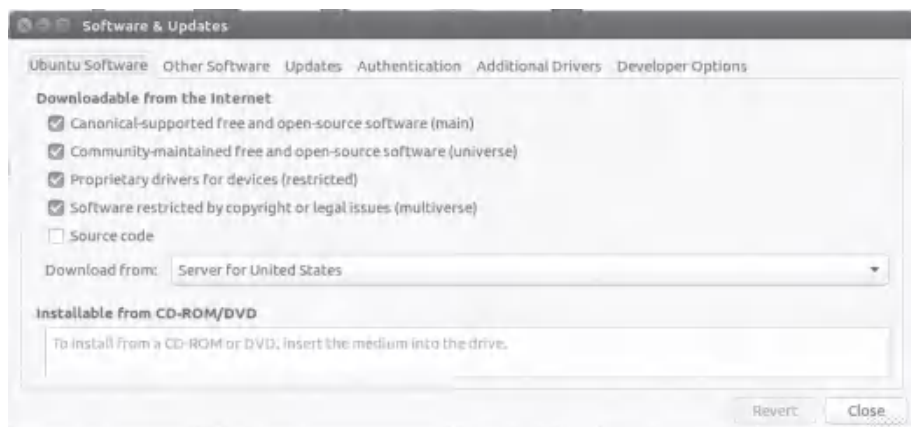


图 1.32 设置 Ubuntu 的软件源

为了安装 ROS 系统，必须选择上面的所有软件源，这样 Ubuntu 就可以从这些软件源中检索安装包了。

2. 通过二进制包安装 ROS

ROS 提供了两种安装方式，第一种是通过二进制包安装 ROS，直接用预先编译完成的二进制文件来安装 ROS，该方法简单、省时，适合使用 ROS 进行应用程序开发的普通用户；第二种是通过源代码编译来安装 ROS，需要先下载 ROS 的源代码，在用户的计算机上编译成二进制文件后再安装，该方法的下载和编译都需要更多的时间，这取决于用户的计算机配置，适合对 ROS 系统有深入了解的高级用户。

本书采用二进制包的安装方法，选择好要安装的 ROS 版本之后，即可在 Ubuntu 系统上安

装 ROS，方法如下。

(1) 安装 Kinetic 桌面完整版，在终端中输入以下命令：

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

安装 Melodic 桌面完整版，在终端中输入以下命令：

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

(2) 安装 Kinetic 桌面版，在终端中输入以下命令：

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

(3) 安装 Kinetic 基础版，在终端中输入以下命令：

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

(4) 单独安装功能包，在终端中输入以下命令（使用所需的功能包名替换命令行中的“PACKAGE”字样）：

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

例如，安装机器人 SLAM 地图建模 Gmapping 功能包时，使用如下命令安装：

```
$ sudo apt-get install ros-kinetic-slam-gmapping
```

若要查找可用的功能包，请运行以下命令：

```
$ apt-cache search ros-kinetic
```

用户根据自己的需要选择其中一种安装方式即可，本书选择 Ubuntu 16.04 + ROS Kinetic 版本的组合。安装的过程中，需要从网络下载很多安装包（二进制的安装文件），耗时可能会很长，需要耐心等待。

1.3.3

配置 ROS

成功安装 ROS 系统之后，需要完成一些配置工作，否则 ROS 系统无法使用。

(1) 初始化 rosdep。

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 核心功能包必须用到的工具。例如，一个 ROS 功能包可能需要若干个依赖包才能正常工作，rosdep 会检测依赖包是否可用，如果不可用，它将自动安装这些依赖包。

在终端中输入以下命令，进行初始化和更新：

```
$ sudo rosdep init
```

```
$ rosdep update
```

(2) 设置环境变量。

此时，ROS 已成功安装在计算机中，默认在 /opt 路径下。在后续使用中，由于会频繁使用终端输入 ROS 命令，所以在使用前需要对环境变量进行设置。

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
```

需要说明的是，本书采用 ROS 的 Kinetic 版本，若为 ROS 其他版本，将代码中的“kinetic”字样替换成对应的版本名称即可。

（3）安装 rosinstall。

rosinstall 是 ROS 中一个独立的、常用的命令行工具，用来下载和安装 ROS 功能包的程序。为了便于后续开发，建议按如下命令安装：

```
$ sudo apt-get install python-rosinstall
```

1.3.4

测试 ROS

1. 启动 ROS

在终端中输入以下命令：

```
$ roscore
```

如果出现如图 1.33 所示的日志信息，说明 ROS 系统安装成功，可以正常启动了。

```
kinetic@kinetic:~$ roscore
... logging to /home/kinetic/.ros/log/87ddad3c-edaa-11ec-9267-000c29aae71b
/roslaunch-kinetic-2679.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is 41GB.

started roslaunch server http://kinetic:40011/
ros_core version 1.12.17

SUMMARY
-----
PARAMETERS
 * /rostdistro: kinetic
 * /rosversion: 1.12.17

NODES
-----
auto-starting new master
process[master]: started with pid [2689]
ROS_MASTER_URI=http://kinetic:11311/

setting /run_id to 87ddad3c-edaa-11ec-9267-000c29aae71b
process[roscout-1]: started with pid [2702]
started core service [/roscout]
```

图 1.33 ROS 启动成功后的日志信息

2. 查看 ROS 的版本

在终端中输入以下命令：

```
$ rosversion -d
```

如果命令行输出的信息是“kinetic”，也说明 ROS 系统安装成功了。

3. 测试 ROS 的吉祥物“小海龟”程序

“小海龟”程序（turtlesim）是一个含有“小海龟”机器人的二维模拟器，可以用来简单地测试 ROS 系统运行是否正常，同时也能借此来体验一下 ROS 的神奇与精彩之处。

启动 roscore 后，重新打开一个终端窗口，输入如下命令，启动仿真器节点：\$ rosruntime turtlesim turtlesim_node

此时，将在一个新打开的窗口中出现一只小海龟。再重新打开一个终端窗口，输入如下命令，启动键盘控制节点：

```
$ rosruntime turtlesim turtle_teleop_key
```

将鼠标聚焦在最后一个终端的窗口中，然后通过键盘上的方向键操作小海龟移动。如果小海龟可以正常移动，并且在屏幕上留下移动轨迹，如图 1.34 所示，说明 ROS 已经成功地安装、配置且正常运行。需要说明的是，本书采用是 ROS Kinetic 版本，不同的 ROS 版本，小海龟的形状和颜色可能会不同，但操作过程是相同的。

至此，ROS 的安装、配置与测试就全部结束了，可以正式开启 ROS 机器人开发及应用的精彩旅程。

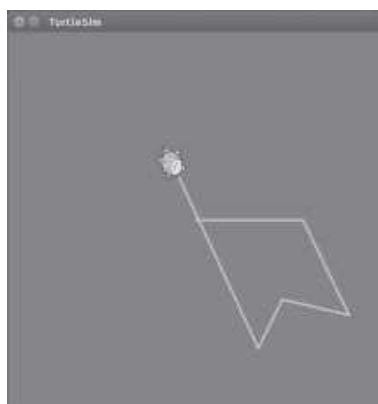
```
kinetic@kinetic:~$ rosversion -d
kinetic
kinetic@kinetic:~$ rosruntime turtlesim turtlesim_node
[ INFO] [1653978798.611510989]: Starting turtlesim with node name /turtlesim
[ INFO] [1653978798.618372562]: Spawning turtle [turtle1] at x=[5.544445], y=[5.544445], theta=[0.000000]
```

(a)启动turtlesim节点

```
kinetic@kinetic:~$ rosruntime turtlesim turtle_teleop_key
Reading from keyboard
-----
Use arrow keys to move the turtle.
█
```

(b)启动键盘控制节点

图 1.34 “小海龟”程序



续图 1.34

机器人

JIQIREN CAOZUO
XITONG JICHU
YU YINGYONG
(DI-ERBAN)



机器人及
人工智能类
创新教材

操作系统(ROS)基础与应用

(第二版)

赛事项目实训手册



AI



机器人操作系统（ROS）基础与运用项目实训手册



赛事实训一 掌握 ROS 核心基础

01

实训题目

ROS 环境搭建与基础操作竞赛

·

02

对接大赛

与全国职业院校技能大赛中机器人应用开发赛项对接，该赛项要求选手具备扎实的机器人操作系统基础环境搭建和基本操作能力。

·

03

大赛要求

知识点：ROS 系统架构与生态的理解，包括核心概念、设计思想、起源与发展；Ubuntu 操作系统与 ROS 的安装与配置方法，如软件源设置、不同版本 ROS 及 Ubuntu 组合的选择。

技能点：熟练掌握 VMware 虚拟机安装与配置；Ubuntu 操作系统在虚拟机及实体机上的安装与设置技巧，如分区操作；ROS 环境部署、验证及常见问题解决，如 rosdep 初始化与更新。

综合素质：培养学生自主学习能力和解决问题的能力，在面对安装配置过程中的错误时能够独立查找资料、分析原因并解决；提高学生的时间管理能力，在规定时间内完成复杂的安装配置任务。

04

大赛准备

学生复习教材中 ROS 核心基础章节内容，重点回顾安装与配置过程中的关键步骤和注意事项。

准备好安装所需的软件资源，如 VMware 软件、Ubuntu 镜像文件、ROS 安装包等，并确保计算机硬件满足安装要求。

05

实训步骤



1. 在规定时间内, 学生在计算机上安装 VMware 虚拟机, 按照教材步骤进行设置, 包括选择合适的安装路径、配置虚拟机硬件参数等。
2. 在虚拟机中安装 Ubuntu 操作系统, 完成分区、设置用户名和密码等操作, 注意选择正确的软件源并安装必要的软件包。
3. 安装 ROS 系统, 根据大赛指定的版本进行安装, 如选择 Kinetic 版本, 完成桌面完整版、桌面版、基础版或单独功能包的安装, 并进行环境变量设置和工具安装, 如 `roscdep` 初始化与更新、`roscinstall` 安装。
4. 启动 ROS 系统, 通过运行 `roscscore` 命令检查安装是否成功, 并查看系统信息, 如节点列表、话题信息等。
5. 运行“小海龟”程序进行测试, 包括启动 `turtlesim` 节点和 `turtle_teleop_key` 节点, 通过键盘控制小海龟移动, 观察小海龟的运动轨迹和系统反馈, 确保整个 ROS 环境运行正常。

06

实训评价



评价点

VMware 虚拟机安装与配置的准确性和完整性; Ubuntu 操作系统安装过程的正确性, 包括分区合理性、软件源设置有效性; ROS 安装的成功率和环境变量设置的正确性; “小海龟”程序测试的运行效果, 如小海龟移动是否流畅、节点通信是否正常。

分层评价

初级: 能够在教师指导下完成大部分安装步骤, 但可能在一些细节设置上出现错误, 如软件源未正确选择导致部分软件包安装失败; 能基本运行“小海龟”程序, 但操作不熟练, 节点通信偶尔出现问题。

中级: 独立完成安装和配置任务, 仅有少量非关键步骤的失误; 熟练运行“小海龟”程序, 能正确理解节点之间的关系, 并能初步分析和解决一些简单的运行问题。

高级: 快速、准确地完成所有安装和配置任务, 且能对安装过程中的一些常见问题进行提前预防 and 解决; 在“小海龟”程序测试中, 不仅能熟练操作, 还能深入分析系统运行机制, 并根据需求对程序进行简单修改和扩展。

07

评价细则

分项	占比	评分细则	赋分
Mware 虚拟机安装	20%	安装路径正确（5%）	
		硬件参数配置合理（10%）	
		虚拟机网络设置正确（5%）	
Ubuntu 操作系统安装	30%	分区方式符合要求（10%）	
		软件源设置正确（10%）	
		用户名和密码设置规范（5%）	
		必要软件包安装完整（5%）	
ROS 安装	20%	安装版本选择正确（5%）	
		安装过程无错误（10%）	
		环境变量设置准确（10%）	
		rosdep 和 rosinstall 安装及操作正确（5%）	
“小海龟”程序测试	20%	小海龟能正常移动（10%）	
		节点启动和通信正常（5%）	
		能对简单问题进行分析和解决（5%）	

08

大赛结果呈现

以学生完成安装和测试的时间、操作的准确性和流畅性为主要呈现指标。可以通过记录学生在每个步骤的完成时间、错误次数等数据进行量化呈现，同时结合学生在操作过程中的表现，如解决问题的能力、对知识的理解深度等进行综合评价，最终给出排名和成绩。

09

大赛反思

分析学生在实训过程中出现的常见错误，如软件源设置错误、环境变量配置问题等，总结原因并针对性地改进教学方法。对于学生普遍存在困难的知识点和技能点，在后续教学中加强讲解和练习，同时鼓励学生在课后进一步复习和实践，提高学生的自主学习能力和实践操作水平。