基于ROS的机器人理论与应用

何炳蔚　张立伟　张建伟　著

科学出版社

基于 ROS 的机器人理论与应用

何炳蔚　张立伟　张建伟　著

科学出版社

北京

内 容 简 介

自 2010 年开源机器人操作系统(ROS)发布第一个版本以来，截至本书成稿时已经发布了 10 个版本，ROS 也已经成为机器人研发领域的通用性软件平台。ROS 是建立在开源操作系统 Ubuntu 系统之上的开源机器人操作系统，其主要目标是为机器人研究和开发提供代码复用的支持。它提供了操作系统应有的服务，包括硬件抽象、底层设备控制、共用功能执行、进程间消息传递，以及包管理。ROS 的官方网站也提供了各种支持文档，相关资源构成了一个强大的生态系统，使学习和使用 ROS 非常方便。本书通过介绍 ROS 并以实际机器人为平台，展示机器人主要功能模块涉及的相关理论和应用场景。

本书可作为机器人研究者和爱好者应用 ROS 构建机器人软件系统的参考手册，也可作为高等院校本科生和研究生的教材。

图书在版编目(CIP)数据

基于 ROS 的机器人理论与应用/何炳蔚，张立伟，张建伟著. —北京: 科学出版社, 2017. 6

ISBN 978-7-03-053057-8

Ⅰ. ①基… Ⅱ. ①何… ②张… ③张… Ⅲ. ①机器人-操作系统-程序设计 Ⅳ. ①TP242

中国版本图书馆 CIP 数据核字 (2017) 第 121611 号

责任编辑: 任 静 / 责任校对: 于佳悦
责任印制: 张 倩 / 封面设计: 迷底封装

科学出版社 出版
北京东黄城根北街 16 号
邮政编码: 100717
http://www.sciencep.com

北京新华印刷有限公司 印刷
科学出版社发行 各地新华书店经销
*
2017 年 6 月第 一 版 开本: 720 × 1000 1/16
2017 年 6 月第一次印刷 印张: 11 1/2
字数: 219 000

定价: 68.00 元

(如有印装质量问题, 我社负责调换)

前　言

截至本书成稿时，开源机器人操作系统 (ROS) 已经发布到第 10 个版本 ROS Kinetic Kame，ROS 也成为机器人研发领域的通用性软件平台。

ROS 具有点对点设计、不依赖编程语言、开源等优点，很快在机器人研究和工业领域得到广泛应用。另外，ROS 官方网站提供了各种支持文档，提供了一套“一站式”的方案，使用户得以搜索并学习全球开发者共享的开源程序包。

作者自 2010 年 ROS 发布伊始，就开始使用 ROS，积累了大量的使用心得和研发经验。全书内容共分 8 章，覆盖了 ROS 和机器人的主要功能模块。在介绍相关使用方法之后，给出在实际通用机器人平台 (如 TurtleBot) 上的运行示例，以方便读者研究和学习。

作者首先感谢为本书的出版提供了意见、支持和帮助的人，感谢福州大学机械工程及自动化学院和德国汉堡大学多模式研究所的同事和同学提供的帮助和支持。同时感谢石进桥、颜培清、易宗超、朱明珠、杨狄赛、吴方熠等协助调试相关章节代码。

本书相关研究获得国家自然科学基金 (项目编号：61673115、61473090)、福建省自然科学基金 (项目编号：2017J01749)、福建省高校新世纪优秀人才支持计划和中德跨区域协同研究中心重大国际合作计划项目 SFB TRR169 (DFG/NSFC funded joint-project “Cross-Modal Learning” under contract Sonderforschungsbereich Transregio 169) 的资助。

由于作者水平有限，书中难免存在不足之处，恳请广大读者和同行批评指正。

目　录

第1章 ROS 简 介

ROS（robot operating system）[1-11] 是一个适用于机器人的开源的元级操作系统。ROS 的主要设计目标是为机器人研发过程中的代码复用提供支持。它提供了操作系统 [12-19]应有的服务，包括硬件抽象、底层设备控制、共用功能执行、进程间消息传递，以及包管理。

ROS 也提供用于获取、编译、编写和跨计算机运行代码所需的工具和库函数。ROS 是一个分布式的进程框架，这些进程被封装在易于被分享和发布的功能包 (package) 中。ROS 也支持一种类似代码储存库的联合系统，这个系统也可以实现工程的协作及发布。这个设计可以使一个项目的开发实现了从文件系统到用户接口的完全独立决策。同时，所有项目都可以被 ROS 的库和基础工具整合在一起。

ROS 相较于其他机器人操作系统主要有以下特点。

(1) 通道：ROS 提供了一种发布-订阅式的通信框架，用以简单、快速地构建分布式计算系统。

(2) 仿真和数据可视化工具：ROS 提供了大量的仿真和数据可视化工具组合，用以配置、启动、自检、调试、可视化、登录、测试、终止系统。

(3) 强大的库：ROS 提供了大量的库文件 (如 roscpp、rospy[20-22])，实现了自主移动、操作物体、感知环境等功能。

(4) 生态系统：ROS 的支持与发展构成了一个强大的生态系统。官方网站 (www.ros.org) 提供了各种支持文档，提供了一套 “一站式” 的方案，使用户得以搜索并学习全球开发者共享的开源程序包。

1.1 ROS 的历史

ROS 系统最早源于 2007 年斯坦福大学人工智能实验室的 STAIR 项目与机器人技术公司 Willow Garage [23] 的个人机器人项目 (personal robotics program) 之间的合作，2008 年之后由 Willow Garage 公司推动其发展。目前，稳定版本有以下几种。

(1) ROS Kinetic Kame，2016 年 5 月 23 日发布 (其 Logo 见图 1.1)。

图 1.1　ROS 版本 Kinetic Kame

(2) ROS Jade Turtle，2015 年 5 月 23 日发布 (其 Logo 见图 1.2)。

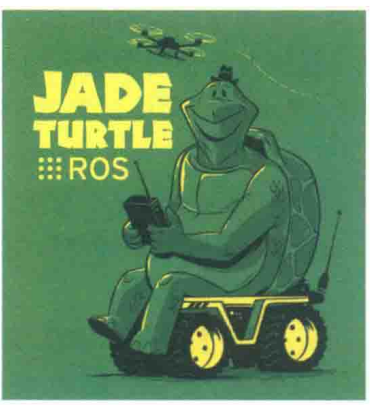

图 1.2　ROS 版本 Jade Turtle

(3) ROS Indigo Igloo，2014 年 7 月 22 日发布 (其 Logo 见图 1.3)。

图 1.3　ROS 版本 Indigo Igloo

(4) ROS Hydro Medusa，2013 年 9 月 4 日发布 (其 Logo 见图 1.4)。

图 1.4 ROS 版本 Hydro Medusa

(5) ROS Groovy Galapagos，2012 年 12 月 31 日发布 (其 Logo 见图 1.5)。

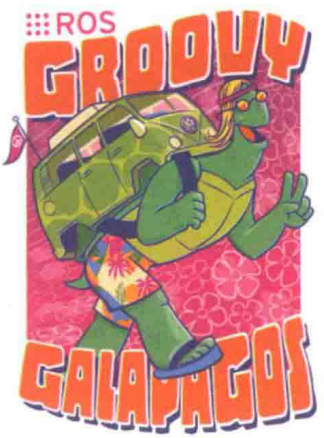

图 1.5 ROS 版本 Groovy Galapagos

(6) ROS Fuerte Turtle，2012 年 4 月 23 日发布 (其 Logo 见图 1.6)。

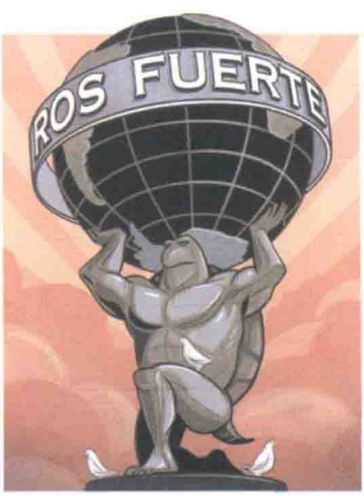

图 1.6 ROS 版本 Fuerte Turtle

(7) ROS Electric Emys，2011 年 8 月 30 日发布 (其 Logo 见图 1.7)。

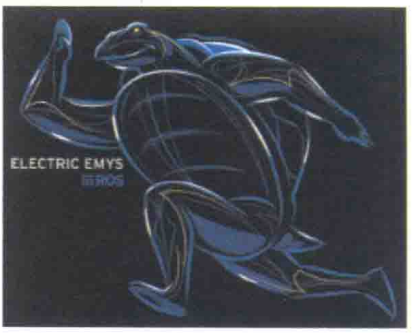

图 1.7　ROS 版本 Electric Emys

(8) ROS Diamondback，2011 年 3 月 2 日发布 (其 Logo 见图 1.8)。

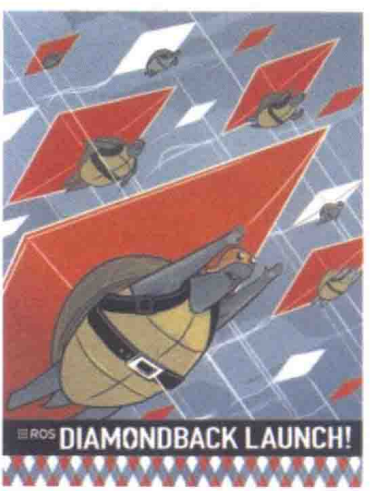

图 1.8　ROS 版本 Diamondback

(9) ROS C Turtle，2010 年 8 月 2 日发布 (其 Logo 见图 1.9)。

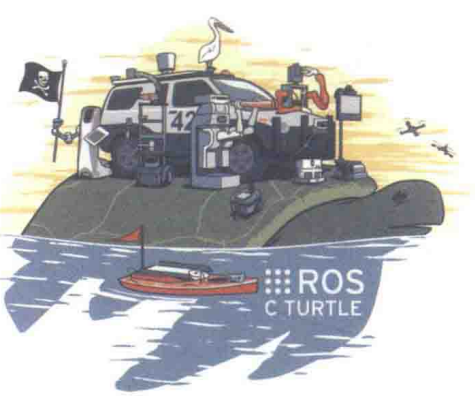

图 1.9　ROS 版本 C Turtle

(10) ROS Box Turtle，2010 年 3 月 2 日发布 (其 Logo 见图 1.10)。

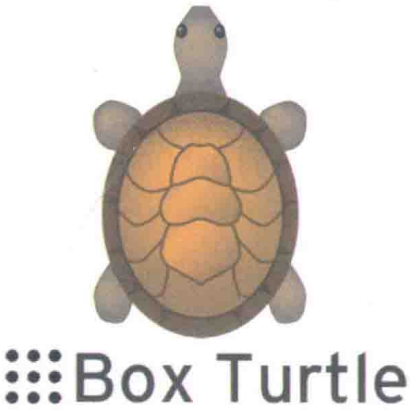

图 1.10 ROS 版本 Box Turtle

1.2 ROS 的 安 装

ROS 目前支持的操作系统有 Ubuntu[24, 25]、OS X、Arch、Federa、Gentoo、OpenSUSE、Slackware、Debian，还可以在 Windows 和 FreeBSD 上安装部分功能。由于 ROS 主要支持 Ubuntu 操作系统，因此，本书以 Ubuntu 操作系统下的安装及使用为例，详细描述 ROS 的安装方法。

1.2.1 ROS Fuerte Turtle 的安装

本小节以 ROS Fuerte Turtle 在 Ubuntu 12.04 LTS[24, 25] 上面安装为例，介绍 ROS 的安装过程。

1) 配置 Ubuntu 系统

配置 Ubuntu repositories 为 “restricted” “universe” 和 “multiverse”。

2) 配置 sources.list

设置计算机使其可以从 ROS.org 接收软件。

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

3) 设置 keys

```
wget http://packages.ros.org/ros.key -O - | sudo apt-key add -
```

4) 安装

重新定向 ROS 服务器：

```
sudo apt-get update
```

下面提供四个版本的安装命令。

(1) 桌面完全版安装 (推荐安装)：ROS、rx、rviz、robot-generic 库、2D/3D simulators、navigation 和 2D/3D perception。

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

(2) 桌面版安装：ROS、rx、rviz 和 robot-generic 库。

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

(3) ROS-Base：ROS 主要功能包、build 和 communication 库，不安装 GUI 工具。

```
sudo apt-get install ros-fuerte-ros-comm
```

(4) 独立的功能包集：用户也可以安装特定的 ROS 功能包集。

```
sudo apt-get install ros-fuerte-STACK
```

例如：

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

5) 环境设置

环境变量设置是为了每次一个新 shell 被调用时，ROS 的环境变量自动被加入到用户的 bash session 中。

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

如果安装了不止一个版本的 ROS，那么 ~/.bashrc 必须是当前使用版本的唯一源 setup.bash。

如果需要修改当前 shell 的环境，可以使用下面的命令：

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

1.2.2 ROS Indigo Igloo 安装

本节以 ROS Indigo Igloo 在 Ubuntu 14.04 LTS[24, 25] 上面的安装为例，介绍 ROS 的安装过程。

1) 配置 Ubuntu 系统

配置 Ubuntu repositories 为 “restricted” “universe” 和 “multiverse”。

2) 配置 sources.list

设置计算机使其可以从 ROS.org 接收软件。

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu \(lsb\_
    release -sc main" > /etc/apt/sources.list.d/ros-latest.list'
```

3) 设置 keys

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

4) 安装

重新定向 ROS 服务器:

```
sudo apt-get update
```

ROS 提供四个版本的安装命令。

(1) 桌面完全版安装 (推荐安装): ROS、rx、rviz、robot-generic 库、2D/3D simulators、navigation 和 2D/3D perception。

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

(2) 桌面版安装: ROS、rx、rviz 和 robot-generic 库。

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

(3) ROS-Base: ROS 主要功能包、build 和 communication 库，不安装 GUI 工具。

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

(4) 独立的功能包集: 用户也可以安装特定的 ROS 功能包集。

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

例如:

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

5) 初始化 rosdep

在使用 ROS 之前，需要初始化 rosdep。

```
sudo rosdep init
rosdep update
```

6) 环境设置

环境变量设置是为了每次一个新 shell 被调用时，ROS 的环境变量自动被加入到用户的 bash session 中。

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

如果安装了不止一个版本的 ROS，那么 ~/.bashrc 必须是当前使用版本的唯一源 setup.bash。

如果需要修改当前 shell 的环境，可以使用下面的命令:

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

第2章 ROS 框 架

通过第 1 章的简单介绍，已经了解了 ROS 的一些基本特点：ROS 是一个开源的元级操作系统 (后操作系统)，一些包、软件工具的集合。它跨机器进行通信的体系架构提供了对系统进行实时数据分析、编程语言独立 (C++、Python、Lisp、Java 等) 等功能。它提供类似于操作系统的服务，包括硬件抽象描述、底层驱动程序管理、共用功能执行、程序间消息传递、程序发行包管理；它也提供一些工具和库，用于获取、建立、编写和执行多机融合的程序。本章将对 ROS 的总体框架、功能包和基本命令进行介绍。

2.1 ROS 总体框架

根据 ROS 系统代码的维护者和分布来标识，ROS 系统代码主要有两大部分。一部分是核心部分，也是主要部分，一般称为 main。主要是由 Willow Garage 公司和一些开发者来提供设计与维护。它们提供一些分布式计算的基本工具，以及整个 ROS 系统核心部分的程序编写。这部分内容被存储在计算机的安装文件中。另一部分是全球范围的代码，被称为 universe，由不同国家的 ROS 社区组织开发和维护。其中包括各种库的代码，如 OpenCV、PCL 等；库的上一层是从功能的角度提供的代码，如人脸识别等，它们调用各种库来实现这些功能；最上层的代码是应用级代码，叫作 apps，可以让机器人完成某一种应用，如去拿啤酒，这个过程则调用不同功能的代码进行组合，如啤酒的识别、抓取啤酒等。这个过程一般需要用户下载相应的功能包，然后学习和使用。

不过，对于使用者来说，无论谁提供设计和维护的代码，用户都可以下载到自己的计算机上，然后进行下一步工作。还可以从另外的角度来理解 ROS。ROS 系统有三级概念：文件系统级、计算图级、社区级。

2.1.1 文件系统级

ROS 文件系统级指的是可以在硬盘上面查看的 ROS 源代码，包括如下几种形式。

(1) 功能包。功能包是 ROS 中组织软件的主要形式，一个功能包可能包含 ROS 运行过程 (如节点)，一个 ROS 依赖库、数据集、配置文件或者组织在一起的任何其他文件。功能包是 ROS 软件的元级组织形式，它可以包含任何内容：库、工具、

可执行文件等。

(2) Manifest。Manifest 提供关于功能包的元数据 (meta data)，包括其许可信息和依赖信息，指定的编程语言信息 (如编译标记)。它是功能包的一种描述。事实上，它最重要的功能是定义功能包之间的依赖关系。

(3) Message(msg) type。消息的描述，定义了 ROS 中发送消息的数据结构，存储在目录 `my_package/msg/MyMessageType.msg` 下。

(4) Service(srv) type。服务的描述，定义了 ROS 中需求和响应的数据结构，存储在目录 `my_package/srv/MyServiceType.srv` 下。

2.1.2 计算图级

计算图级 (见图 2.1) 是 ROS 处理数据的一种点对点的网络形式。程序运行时，所有进程及它们所进行的数据处理，将会通过一种点对点的网络形式表现出来。它们将通过节点、节点管理器、主题、服务等来进行表现。ROS 中基本的计算图级概念包括节点、节点管理器、参数服务器、消息、服务、主题和包。这些概念以各种形式提供数据。

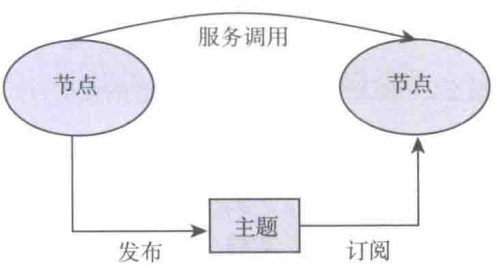

图 2.1 ROS 计算图级概念

一些基本的 ROS 概念如下。

(1) 节点。ROS 节点是用 ROS 客户端库 (如 roscpp、rospy) 写成的执行计算的过程。一个机器人控制系统由很多节点组成，以便在很精细的尺度上模块化。例如，可以通过一个节点进行人脸识别，一个节点执行导航，一个节点进行抓取。

(2) 节点管理器。节点之间通过节点管理器进行名称注册和查找。没有节点管理器，节点将不能互相通信或者进行消息交换。

ROS 节点管理器为节点保存主题和服务的注册信息。节点通过与节点管理器通信来报告其注册信息。当这些节点和节点管理器通信时，它们可以接收别的注册节点的信息，并保持通信正常。当这些注册信息改变时，节点管理器也会回调这些节点。节点可以与节点直接相连。节点管理器仅提供查找表信息，如 DNS 域名服务器。订阅一个主题的节点将会请求与发布主题的节点进行连接，并确定在一种连接协议上进行连接。

(3) 参数服务器。参数服务器是节点管理器的一部分。

(4) 消息。一个消息是一个由类型域构成的简单的数据结构。消息可以包含任何嵌套的结构和阵列。节点之间通过消息来互相通信。

(5) 主题。消息通过主题进行传送。一个节点通过把消息发送到一个给定的主题来发布一个消息。主题是用于识别消息内容的名称。一个节点对某一类型的数据感兴趣，它只需要订阅相关的主题即可。一个主题可能同时有很多并发主题发布者和主题订阅者，一个节点可以发布和订阅多个主题。通常情况下，主题发布者和主题订阅者不知道对方的存在。当订阅者发现该信息是它所订阅的时，就可以在工作区接收到这个信息。

ROS 中有多个独立的节点，节点之间通过一个发布/订阅的消息系统与其他节点联系。如图 2.2 所示，发布者和订阅者都可以是节点，当一个节点需要广播消息时，它就会发布消息到对应的主题。当一个节点想要接收信息时，它可以订阅所需要的主题。

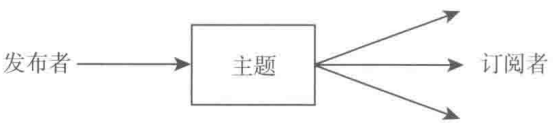

图 2.2　ROS 中发布者和订阅者的通信方式

(6) 服务。发布/订阅模式这种多对多的传输方式不同于请求/回复交互的方式，请求/回复交互的方式通过服务来进行。其中，服务被定义为一对消息结构：一个用于请求，一个用于回复。节点提供了某种名称的服务，客户通过发送请求信息并等待响应来使用服务。

图 2.3 所示的服务是一个客户端节点发送 “请求” 的数据到一个服务器节点，并等待回复；服务器节点接收到 “请求” 后，发送一些称为 “回复” 的数据给客户端节点。“请求” 和 “回复” 数据携带的特定内容由服务数据类型来决定，类似消息的消息类型，但是服务数据类型分别表示请求和回复。服务与消息的不同之处在于：服务是双向的一对一通信，而消息是单向的一对一或者一对多通信。

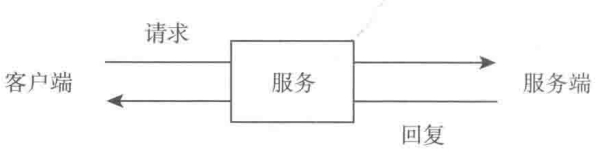

图 2.3　ROS 中客户端和服务端的通信方式

(7) 消息记录包。消息记录包是一种用于保存和回放 ROS 消息数据的格式，是用于检索机器人数据的重要机制。