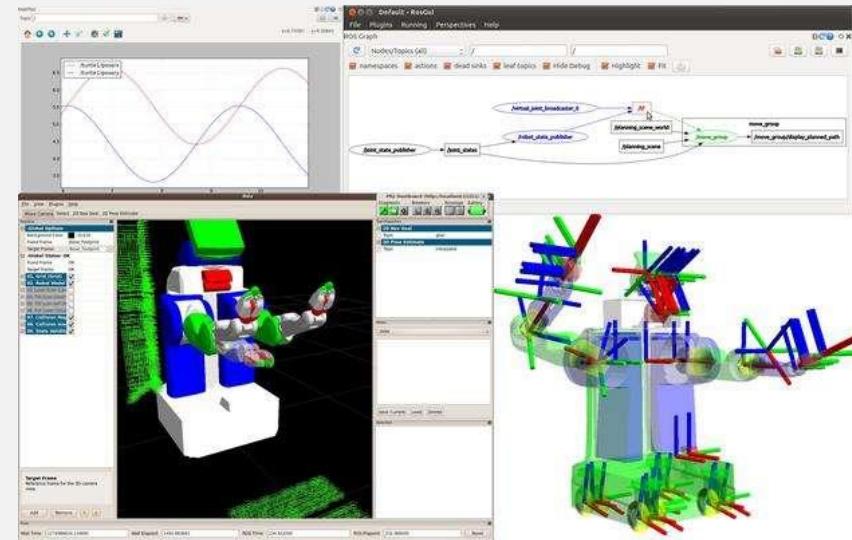


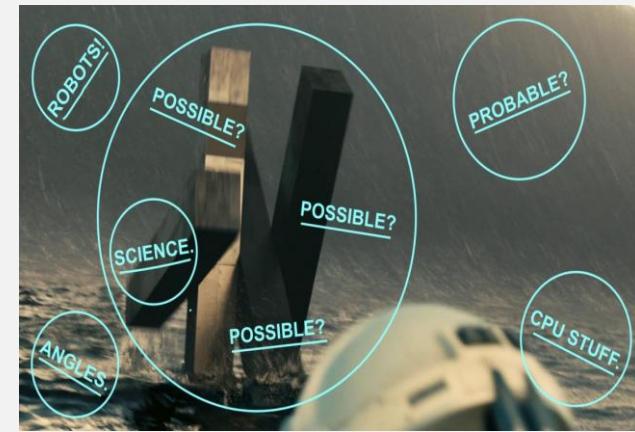
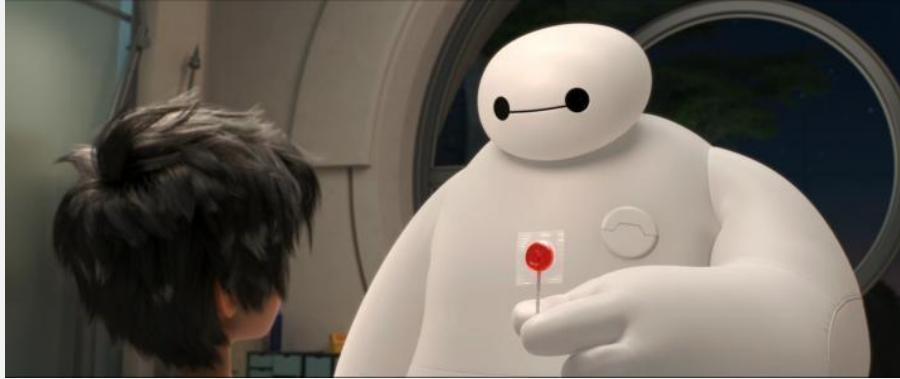
Chapter 1 Introduction to ROS

Wende Ke

Department of Mechanicals and Energy Engineering, SUSTECH



科幻故事中的机器人发展



机器人：国内现状



李克强总理感受百度机器人小度



优必选机器人登上央视春晚



科沃斯扫地机器人改变家庭清洁方式

机器人：国际现状



Google



Boston Dynamics



大疆 Matrice 100



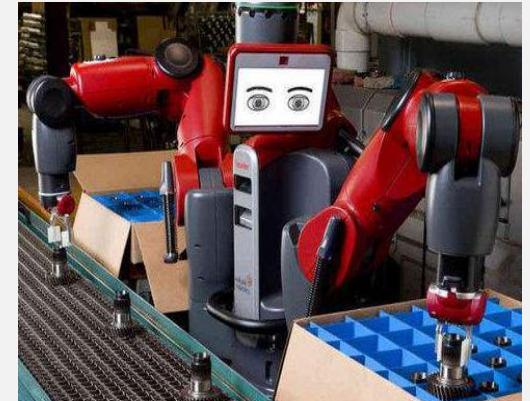
Fetch Robotics-Fetch/Freight



NXROBO-BIG i



Willow Garage-PR2



Rethink Robotics-Baxter

什么是机器人

诞生

百度百科

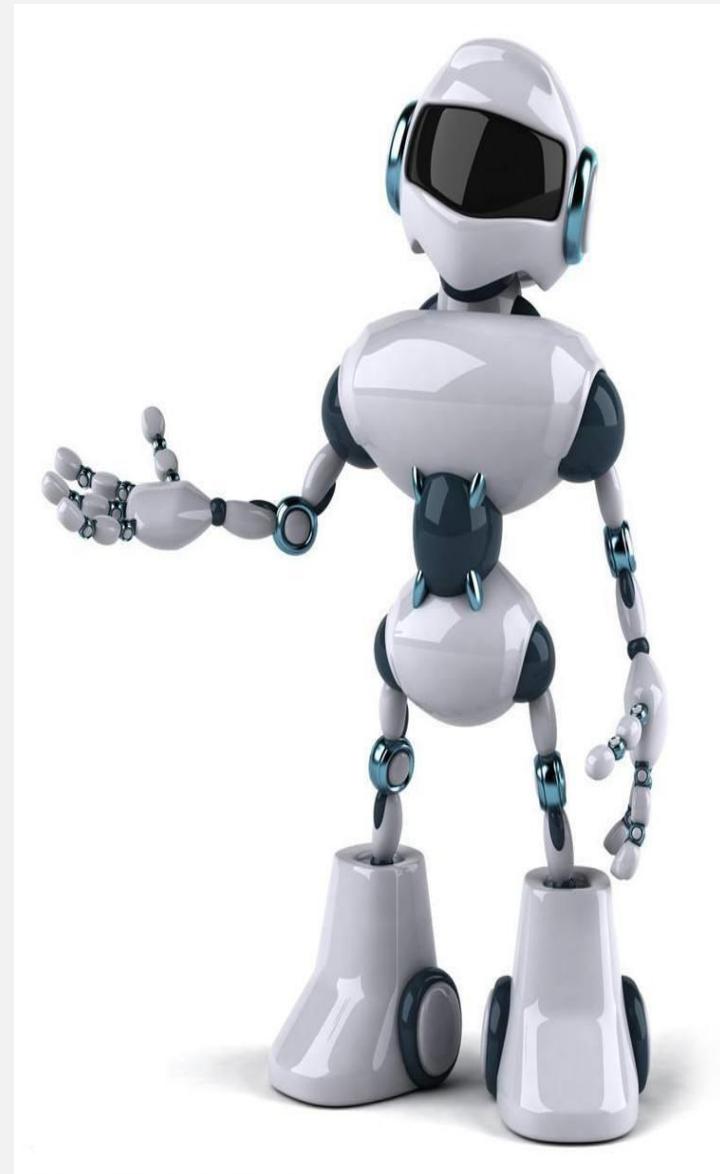
美国机器人协会

1920年捷克斯洛伐克作家卡雷尔·查佩克在他的科幻小说《罗萨姆的机器人万能公司》中，根据Robota(捷克文，原意为“劳役”、苦工)和Robotnik(波兰文，愿意为“工人”)，创造出“机器人”这个词

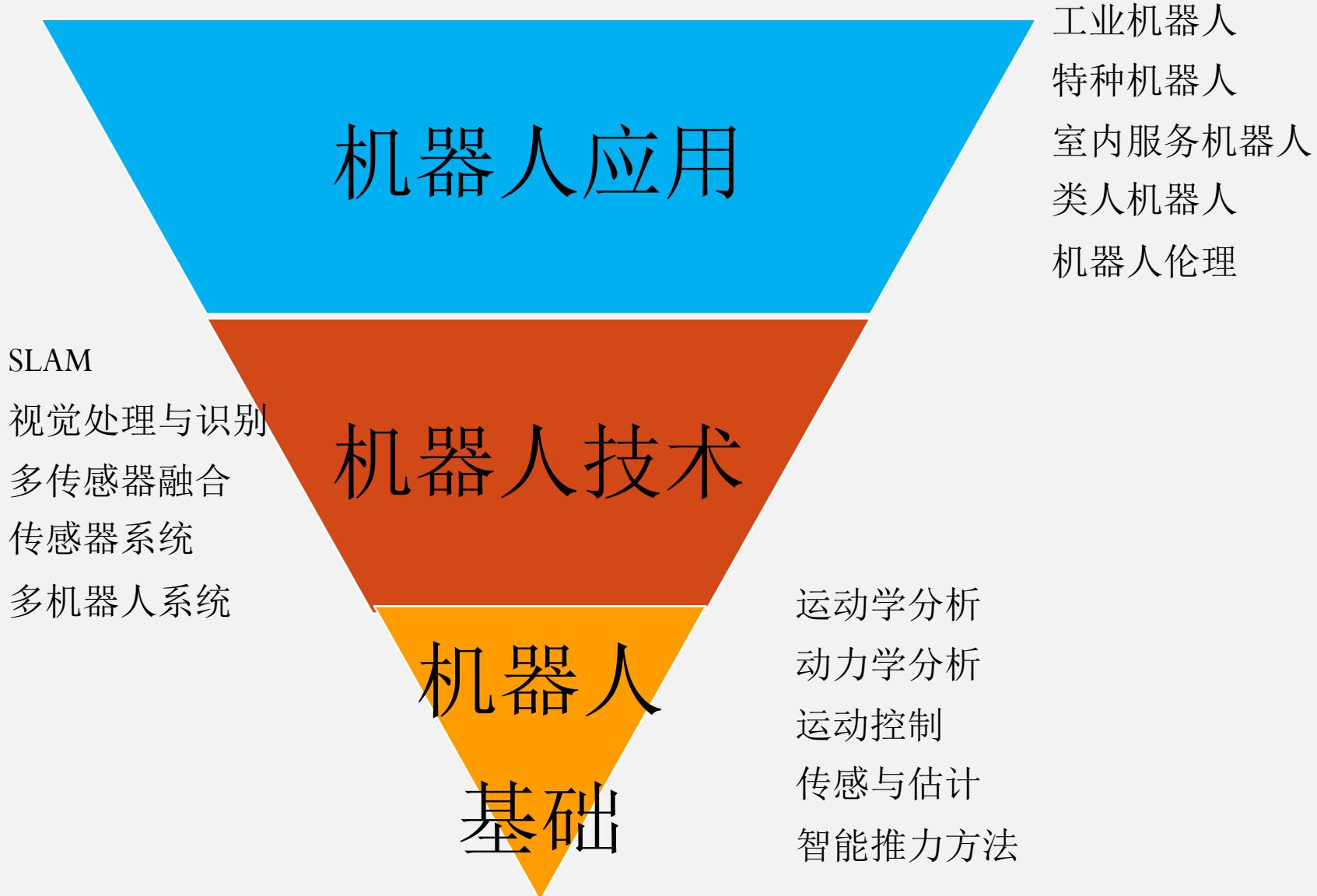
机器人(Robot)是自动执行工作的机器人装置。它既可以接受人类指挥，又可以运行预先编排的程序，也可以根据以人工智能技术制定的原则纲领行动。它的任务是协助或取代人类工作，例如生产业，建筑业，或是危险的生产。

机器人是以搬运材料、零件、工具的可编程的多功能操作器或是通过可改变程序动作来完成各种作业的特殊机械装置。

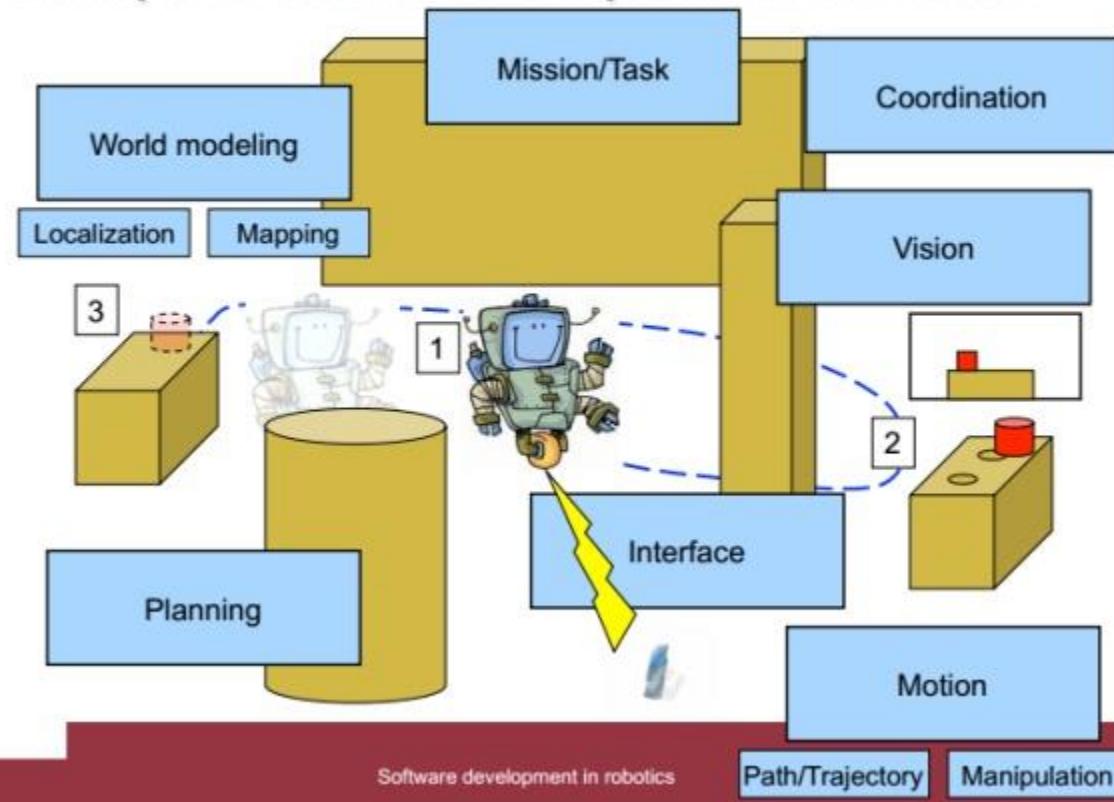
机器人工程是跨学科工程



机器人知识架构



Example of module decomposition in robotics



机器人的普及

- 2007年1月，比尔·盖茨在《科学美国人》上撰文预言：**机器人即将重复个人电脑崛起的道路，走进千家万户**
 - 机器人行业现今面临的挑战，和30年前电脑行业遇到的问题“**如出一辙**”
 - 流行的应用程序很难在五花八门的装置上运行
 - 在一台机器上使用的编程代码，几乎不可能在另一台机器发挥作用，如果想开发新的产品，通常得从零开始
 - 原因
 - 硬件：**标准化工作未开始**
 - 软件：没有**操作系统**
- 媲美30年前的一篇文章：1977年9月Intel公司创始人罗伯特·诺伊斯撰文预言**计算机将走进千家万户**。

SCIENTIFIC AMERICAN

JANUARY 2007

WWW.SCIAM.COM

DAWN OF THE AGE OF ROBOTS

Bill Gates writes that
every home will soon have
smart mobile devices



If This Is a
PLANET,
Then Why
Isn't Pluto?



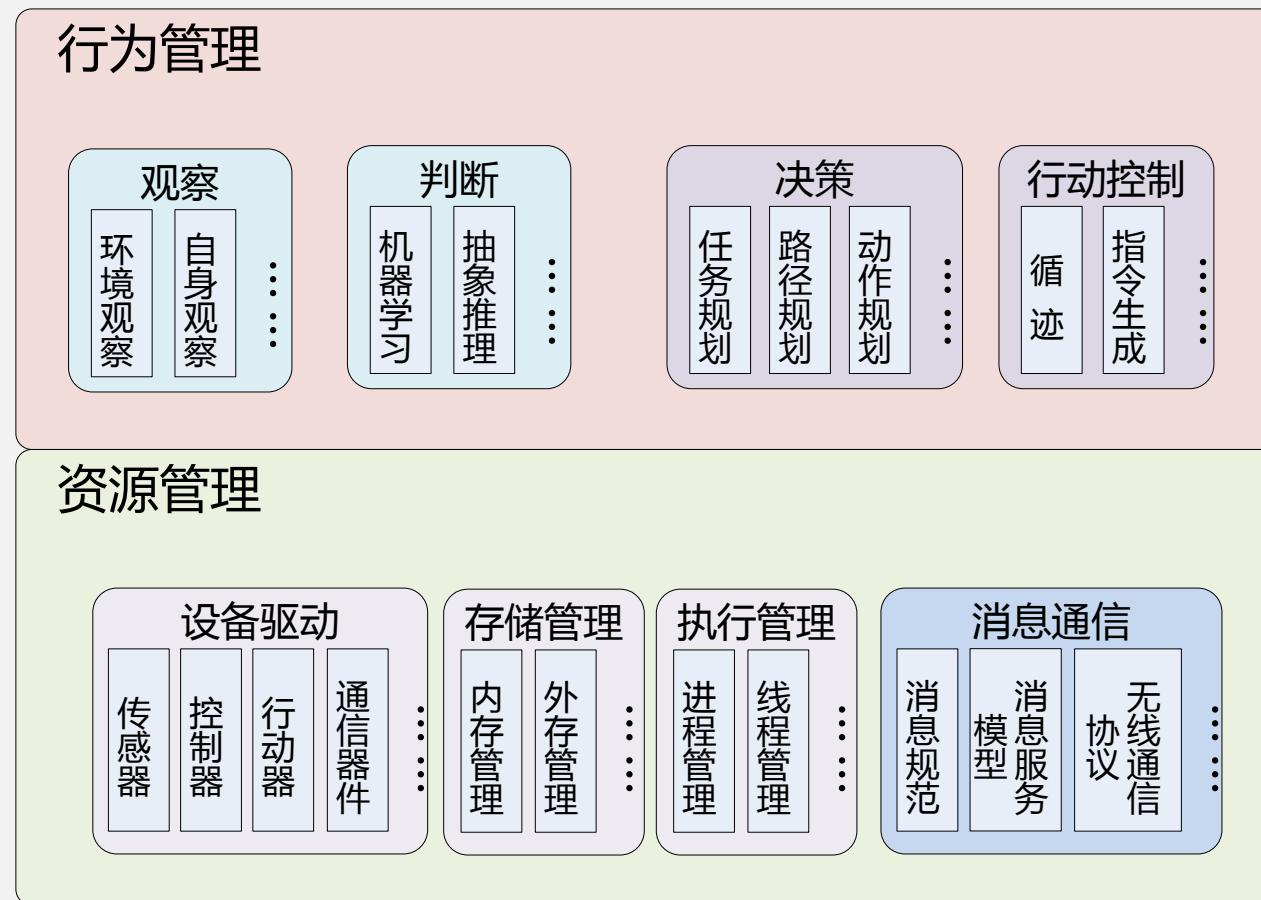
机器人操作系统需要考虑什么？

“Robotics is really a **software** problem. It is not a hardware problem.”

——Google架构师、ROS创始人Scott Hassan

特点一 软件架构

- 纵向看为两层结构：资源管理层、行为管理层



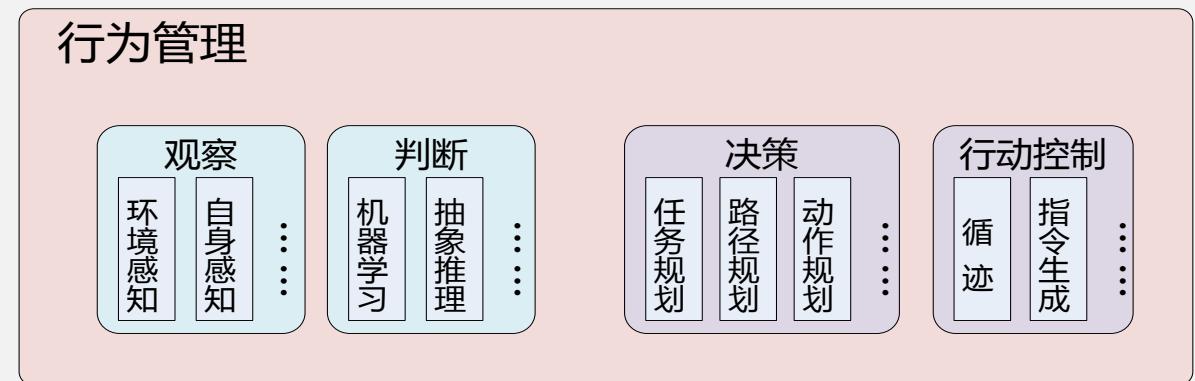
特点一 软件架构

● 软件架构——资源管理层

- 管理与控制机器人硬件资源，屏蔽机器人硬件资源的异构性，并以优化的方式实现对硬件资源的使用
 - 处理器、存储器
 - 通信设备、各类传感器、行为部件等外设
- 管理机器人软件资源，实现软件的部署、运行和协同
- 管理数据的传输、存储和处理
- 提供人机交互接口

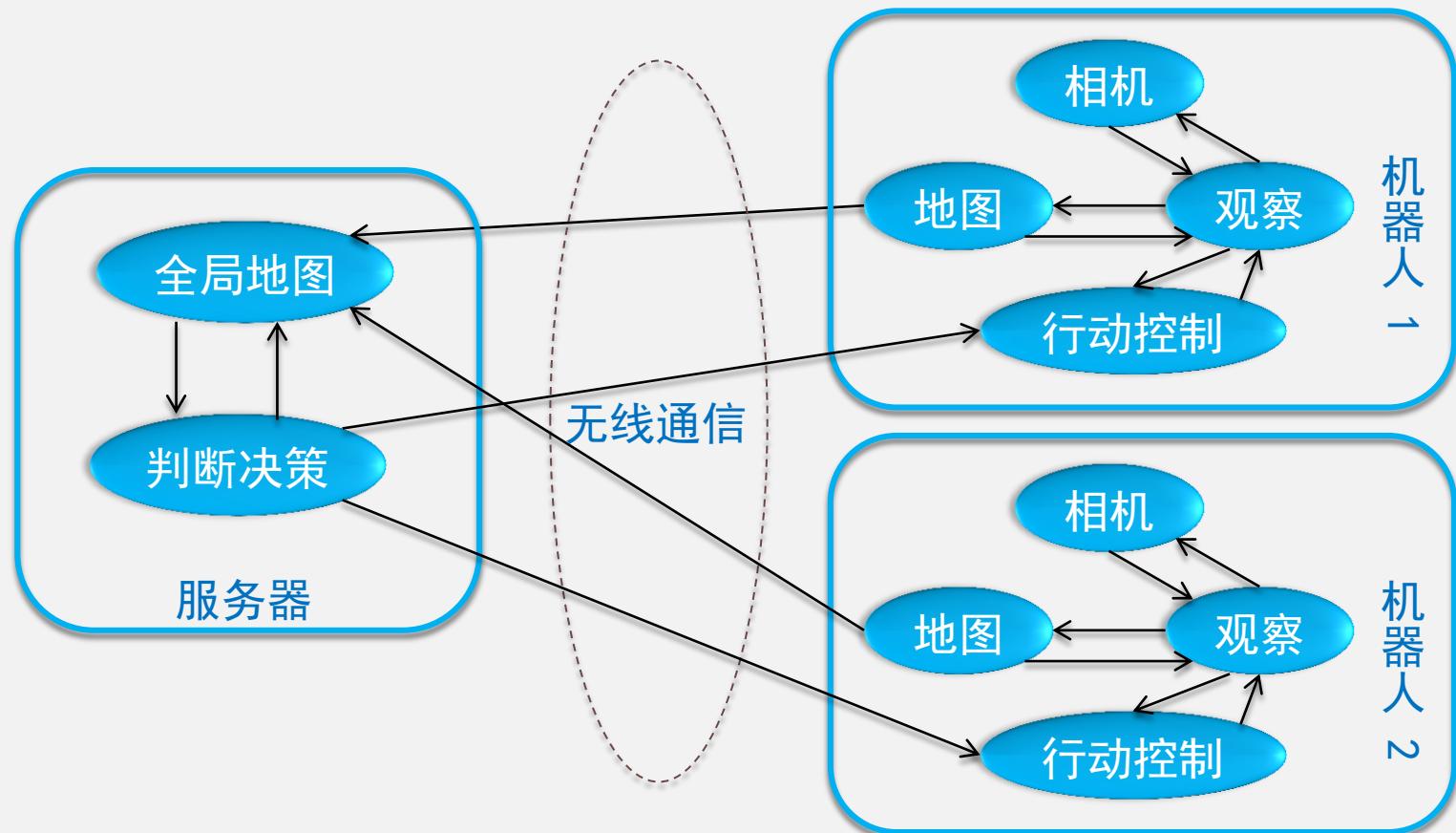


- 软件架构——行为管理层
 - 管理与控制机器人的高级认知（例如观察、判断、决策），并将其转化为作用于物理世界的行动
 - 观察
 - 判断
 - 决策
 - 行动



- 软件架构——横向上的**分布式**结构
 - 机器人的软硬件模块构成**分布式**结构
 - 传感器节点
 - 摄像机、激光扫描测距仪、GPS、惯性测量单元、声呐等
 - 计算存储通信节点
 - 运行判断、规划决策等算法
 - 地图、知识库等
 - 无线通信模块、消息等
 - 控制执行节点
 - 对机械臂等执行部件的行动控制
 - 多机器人也构成**分布式**结构
 - 多个异构的机器人节点
 - 后台服务器节点等

- 软件架构——横向上的**分布式**结构
- 一个典型的机器人操作系统案例



- 运行机制——执行“观察—判断—决策—行动控制”闭环行为链
 - 通过传感器观察环境和自身状态
 - 根据观察，形成判断
 - 进行决策，产生行动方案
 - 控制行动的过程

- 功能
 - 资源管理
 - 管理软硬件、数据资源
 - 满足传感器驱动、行动控制、无线通信、分布式构架等机器人的特殊要求
 - 行为管理
 - 实现行为的抽象和管理，支撑行为的智能化
 - 管理“观察—判断—决策—行动控制”闭环链的调度执行
 - 提供可复用的共性基础软件库和工具
 - 满足行为的可靠性（dependency）约束
- 人机交互方式
 - 输入
 - 任务、环境、自身状态
 - 输出
 - 机器人的行动

特点二——分布架构

基本解决思路

结点实时性

- 通过结点自身计算资源的调度保证实时性
- 向上层应用提供面向机器人领域的实时能力抽象

消息实时性

- 在网络协议层引入支持实时的协议栈（如RT-NET）
- 在应用层提供消息的实时性支持

任务实时性

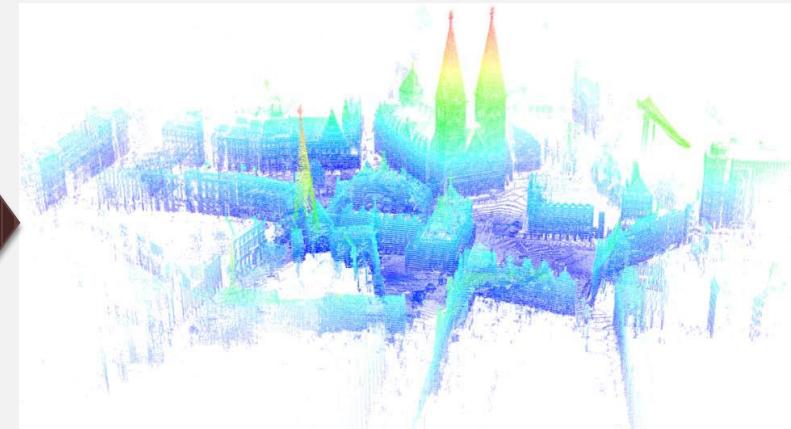
- 提供实时约束在不同结点和信道之间传播的机制
- 具有时间约束的观察——判断——决策——行动控制

特点三——观察与信息融合

- 问题与挑战
 - 环境的表示
 - 环境的观察
 - 传感器信息融合

- 环境的表示

- 机器人**世界模型**的共性化、模块化、标准化
 - 共性、通用、一致的多种世界模型
 - 面向不同的行为与应用场景
 - 针对多传感器、多机器人信息
 - 环境特征和世界模型的数据库



(2013)

- 环境的观察
 - 世界模型的构建和更新
 - 尤其是动态条件下的可靠实现
 - 机器人定位和自身状态的监测
 - 利用外部信息的定位和自主定位
 - 节点的工作状态和本体物理安全状态的监测

- 传感器信息融合
 - 异构传感器的硬件抽象与消息格式标准化
 - 高精度、鲁棒的多传感器信息融合算法库
 - 多机器人协同观察——信息筛选机制

计算机操作系统



机器人操作系统



特点四——机器学习与判断

- 问题与挑战

- (1) 基于大数据与传统人工智能相结合的判断

- (2) 基于机器学习的判断

- 复杂环境下的学习模型
 - 增量式与逐步精化的学习方法
 - 模式识别

具有人类的判断能力是机器人学追求的目标

特点五——规划与决策

- 问题与挑战
 - 面向复杂环境和复杂任务的规划与决策
 - 复杂环境——开放、非结构化、动态、非确定
 - 复杂任务——具有高自主性要求
 - 面向不确定性的可靠规划与决策
 - 面向多机器人协同的规划与决策

- 面向复杂环境和复杂任务的规划与决策
 - **领域无关规划决策库**——基于抽象的、通用的动作模型和系统状态模型
 - 灵活性强、复用性好
 - 问题描述与求解解耦，可以应用多种成熟的规划算法
 - **领域相关规划决策库**——使用专门模型描述需要规划的动作类型和系统状态
 - 针对性强、效率较高
 - **规划决策算法的复合机制与接口**——领域相关规划决策与领域无关规划决策相结合，实现复杂环境中复杂任务的规划与决策

• 面向不确定性的可靠规划与决策

- 针对存在感测不确定性的规划问题，采用非确定性模型与概率模型，如
 - 采用马尔可夫决策过程
 - 采用基于模拟的框架，将增强学习与规划、行动进行结合
- 规划失败时，提供可靠的恢复机制，如
 - 旋翼无人机的紧急拉升
 - 地面移动机器人的旋转恢复动作

- 面向多机器人协同的规划与决策
 - 多智能体机器人系统（MARS）
 - 每个成员仅具有不完全的信息处理和问题求解能力
 - 采用分布式控制
 - 规划过程是异步、并发的
 - 例如，德国人工智能研究中心（DFKI）的“分布式机器人系统集成式任务规划”（IMPERA）使用标准化、模块化的任务规划架构解决异构多机器人的协同规划与决策问题



特点六——行动与控制

- 问题与挑战

- 机器人行为柔性自主控制

- 机器人在行动和执行过程中需要实现不同自主等级的控制，以适应环境的动态变化以及响应人不同程度的人工干预
 - 挑战在于自主控制等级的柔性调节

- 机器人协同行为的一致性控制

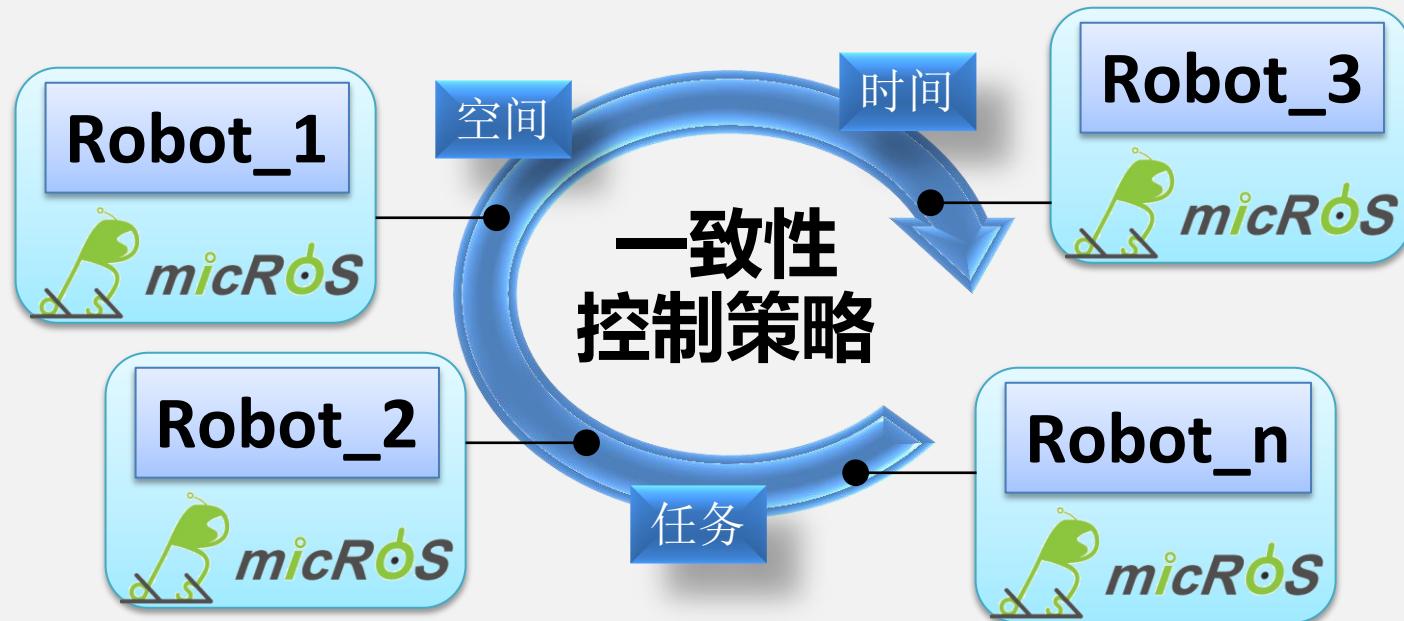
- 协同工作的多个机器人或执行部件，必须达成协调一致行动
 - 难点在于分布式网络条件下的时间、空间和任务协调

- 机器人可变自主权限管理与控制机制
 - 自主权限的表征——自主等级划分与表示
 - 支持不同自主等级控制方法设计，刻画相应的适用条件
 - 提供不同自主等级控制的转换管理，根据环境/任务等各种条件进行自主等级选择

机器人可变自主权限管理与控制

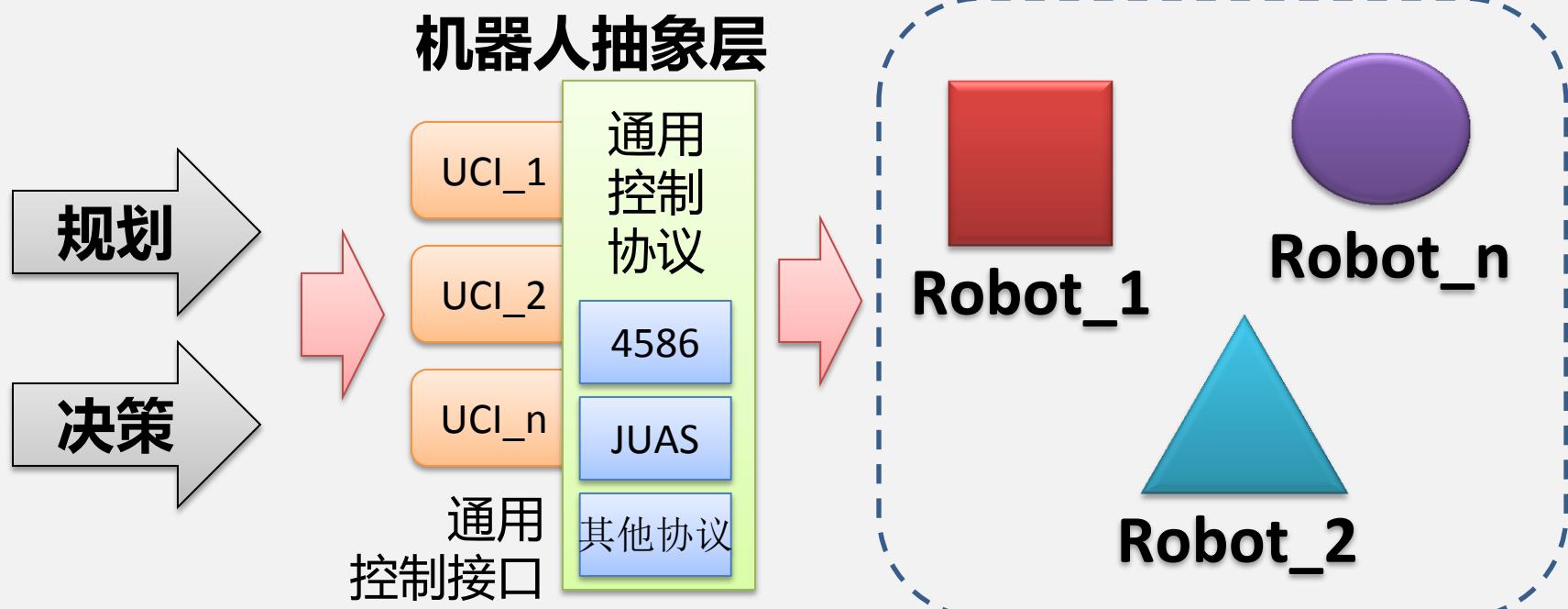


- 分布式协同一致性控制机制
 - 提供一致性控制策略，依据控制拓扑和网络动态特性进行跟踪和预测
 - 机器人队形/构型保持和变换，多机器人合作式避碰



- 面向行为的通用控制接口

- 定义行为层次的统一控制接口，适应不同类型、不同任务的机器人
- 提供多机器人、机器人与人之间的通用互操作控制协议



与传统计算机操作系统的区别

	计算机操作系统	机器人操作系统
架构	分核内/核外两层，实现资源管理	分资源管理、行为管理两层
运行机制	进程/线程模型	观察—判断—决策—行动控制的执行链
功能	管理硬件、软件、数据等资源	除传统资源外，还管理行为
人机交互	输入数据和算法，反馈数据结果	观察环境和自身状态，反馈行动
从应用角度看	工具——实现各种各样信息处理	用具——面向领域执行特定任务
协同	互连、互通、互操作	互操作、互理解、互遵守
关注点	可管理性、实时性、可用性、安全性等	+ 自主性、生存性、对抗性等



- 微软的机器人开发平台RDS
(Robotics Developer Studio)

- 目标是开发不同机器人硬件平台的应用程序
 - 与机器人或控制计算机平台的Windows配合
- 2006年12月推出第一版，最新版本RDS 4为2012年3月发布
 - 不开放源代码，但可免费下载
- 有60家以上的硬件/软件厂商支持或使用该软件开发工具
 - 例如乐高公司

- 开源的机器人软件项目
 - 开源机器人基金会的ROS
 - 美国的Player/Stage
 - 欧洲的Orocос
 - 欧洲的YARP
 - 日本的OpenRTM-aist
 -



The Orocос Project
Smarter control in robotics & automation!





- ROS(Robot Operating System)
 - 起源于2007年Stanford大学AI实验室与Google合作的项目，2008年起由Willow Garage公司维护，2013年起移交开源机器人基金会（OSRF）管理
 - 最初动机是提高代码的可重用率
 - 构建一个能够整合不同研究成果，实现算法发布、代码重用的机器人软件平台
 - 目前包含了3000多个机器人平台的常用软件包
 - 涵盖了硬件驱动、模拟仿真、运动规划、运动控制、环境感知等各个方面

- ROS (Robot Operating System)
 - 正在逐步成为机器人研发领域的**事实标准**
 - 被大学和研究机构广泛采用
 - 学术界指定的创新验证平台
 - DARPA资助的项目和竞赛的平台
 - 已经应用于多种地面/空中/水面/水下无人平台
 - 逐渐向工业领域扩展
 - 对ABB, Adept, Fanuc, Motoman, Universal Robots公司的产品提供支持

“Since version 1.0 was released in 2010, ROS has become the *de facto* standard in robotics software.”

——《MIT Technology Review》, 2013

ROS是什么？

ROS是面向机器人的开源的元操作系统。

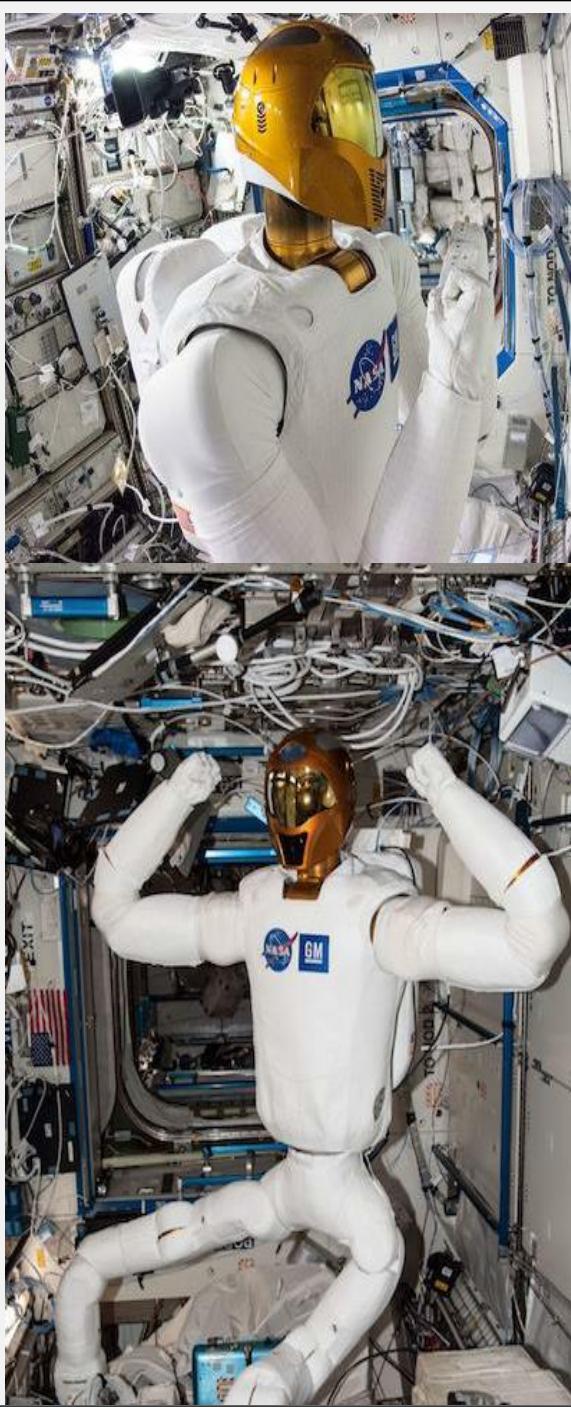
能够提供类似传统操作系统的诸多功能，如硬件抽象、底层设备控制、常用功能实现、进程间消息传递和程序包管理等。

此外，还提供相关工具和库，用于获取、编译、编辑代码以及在多个计算机之间运行程序完成分布式计算。



ROS发展现状

PR2机器人—ROS
中的元老机器人



ROS发展现状
NASA 将运行ROS的
Robonaut2 部署到空
间站

ROS发展现状

机器人领域的事实标准



Comau NM45

Fanuc m10ia

BioRob Arm

KUKA LWF



KUKA OmniROB



Hoap3



extross Hand Aldebaran Romeo CKBot Denso Robot (v.

extross Hand



Aldebaran Romeo

CKBot



Denso Robot (v.)



Universal Robots
UR5/(UR10)



Kinova Jaco



ABB



da Hiro Robonik XL-Terabot HRP 4 Pioneer P3AT

da Hiro



Robonik XL-Terabot



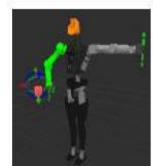
HRP 4



Pioneer P3AT



Research Robot



BDI Atlas



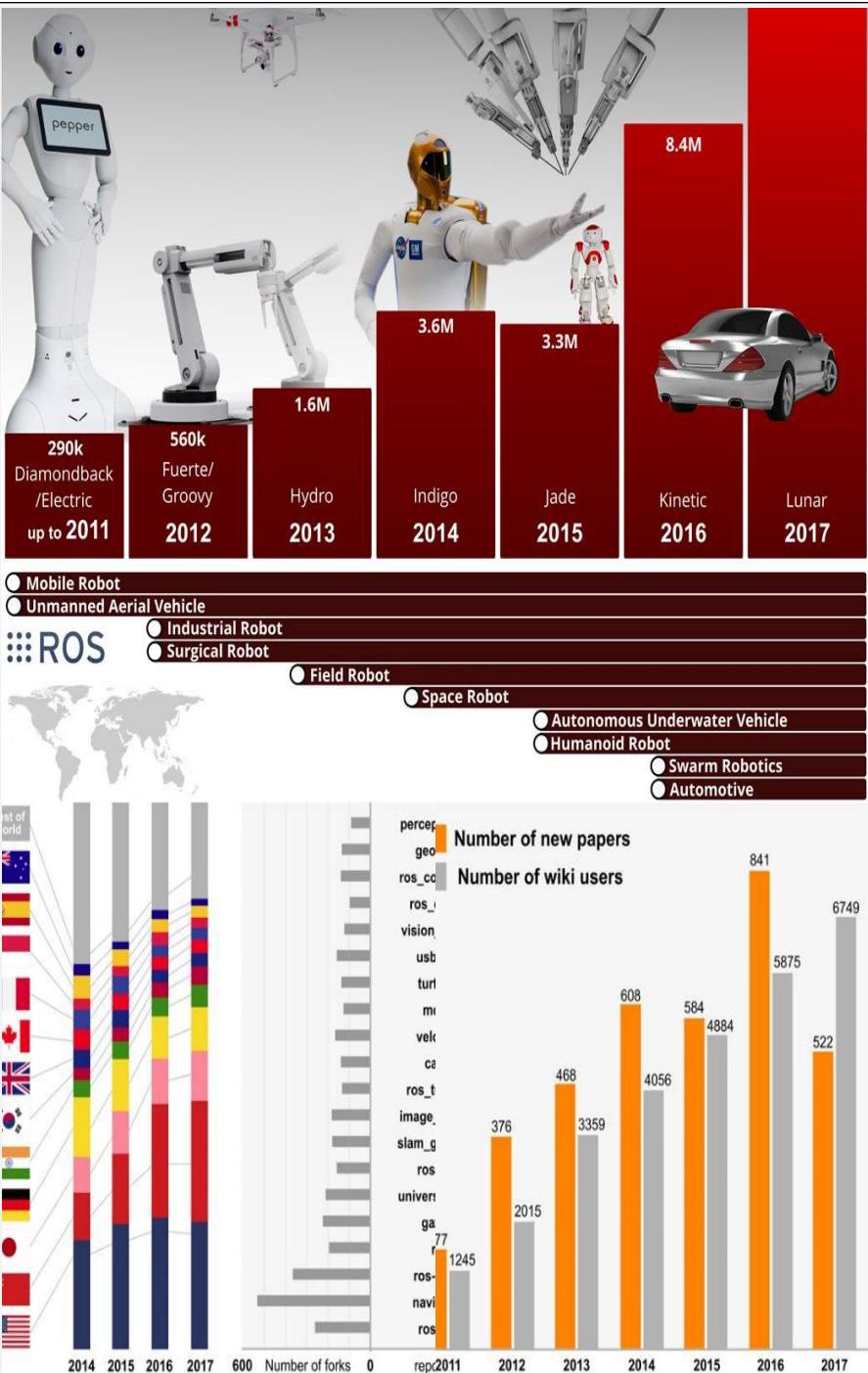
Robona



Aldebaran NAO Care-O-Bot HRP-2



HRP-2



ROS发展现状
ROS社区内的功能包
数量、关注度、相关
文章均呈指数增长。

国家政策

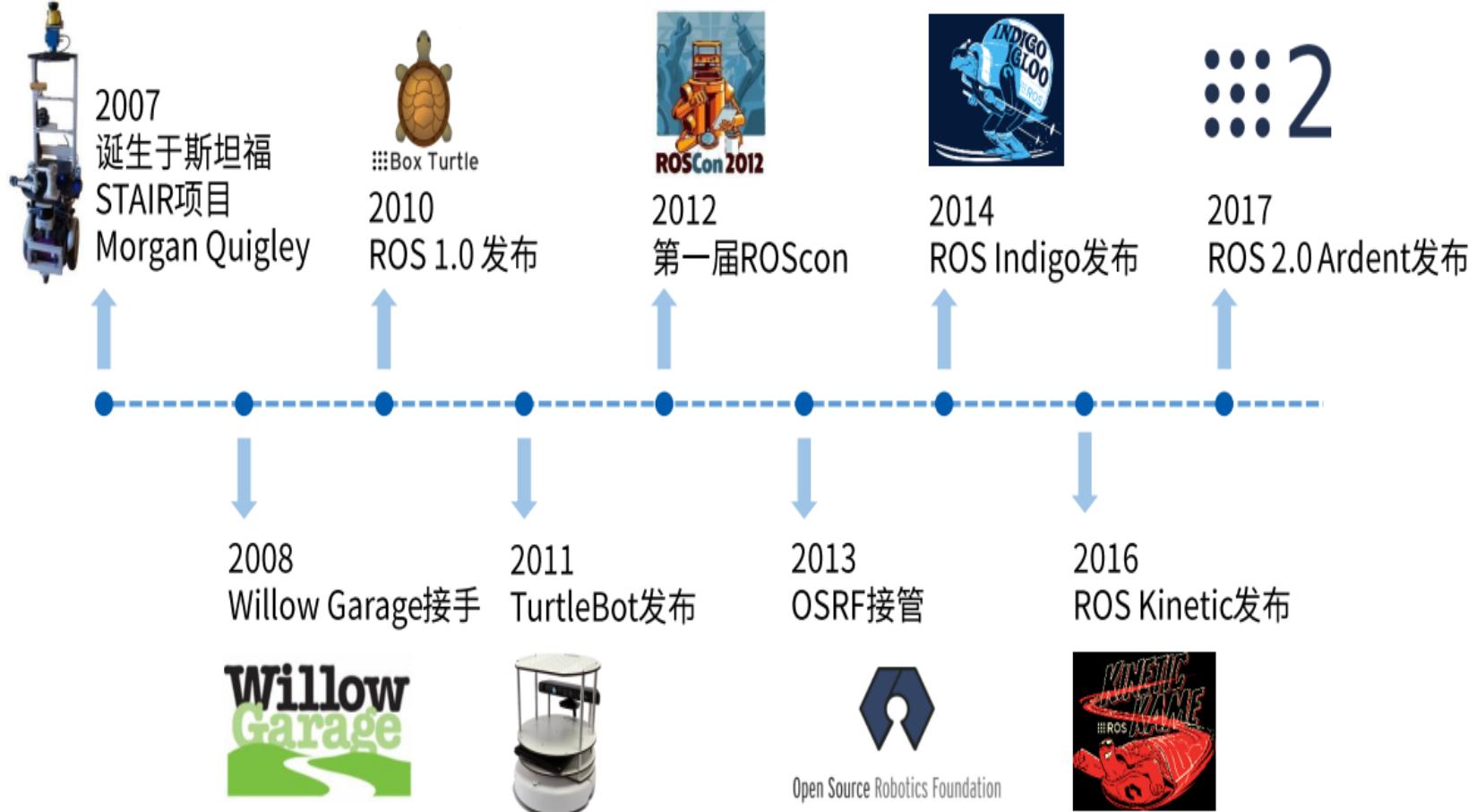
- | | |
|------|---|
| 2020 | 人工智能总体技术要与世界先进水平同步、人工智能产业成为我国新的重要增长点。 |
| 2025 | 人工智能基础理论实现重大突破、技术与应用部分达到世界先进水平，智能社会建设要取得积极进展。 |
| 2030 | 我国成为世界主要人工智能创新中心。 |

2017年国务院印发的《新一代人工智能发展规划》

国家政策



2018国家工作报告提出，加强新一代人工智能研发应用，发展智能产业，拓展智能生活。



ROS 历史起源

ROS Distribution Releases

Distro	Release date	Poster	Tuturtle, turtle in tutorial	EOL date
ROS Kinetic Kame (Recommended)	May 23rd, 2016			May, 2021
ROS Jade Turtle	May 23rd, 2015			May, 2017
ROS Indigo Igloo	July 22nd, 2014			April, 2019 (Trusty EOL)
ROS Hydro Medusa	September 4th, 2013			May, 2015

ROS总体设计目标

提高机器人开发中的代码复用率



传统模式：重复造轮子，效率低下

现代模式：分工合理，重复利用

What's ROS ?

- <http://wiki.ros.org/ROS/Introduction>
- ROS 是一个适用于机器人的开源的元操作系统。
 - 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，常用函数的实现，进程间消息传递，以及包管理。它也提供用于获取、编译、编写、和跨计算机运行代码所需的工具和库函数。
 - 松耦合点对点进程网络。
- 支持的操作系统
 - ROS目前只能在基于Unix的平台上运行。ROS的软件主要在Ubuntu和Mac OS X 系统上测试，同时ROS社区仍持续支持Fedora, Gentoo, Arch Linux和其它Linux平台。
 - 与此同时，Microsoft Windows端口的ROS已经实现，但并未完全开发完成。

ROS的优势

1，松散耦合的机制方便机器人软件框架的组织

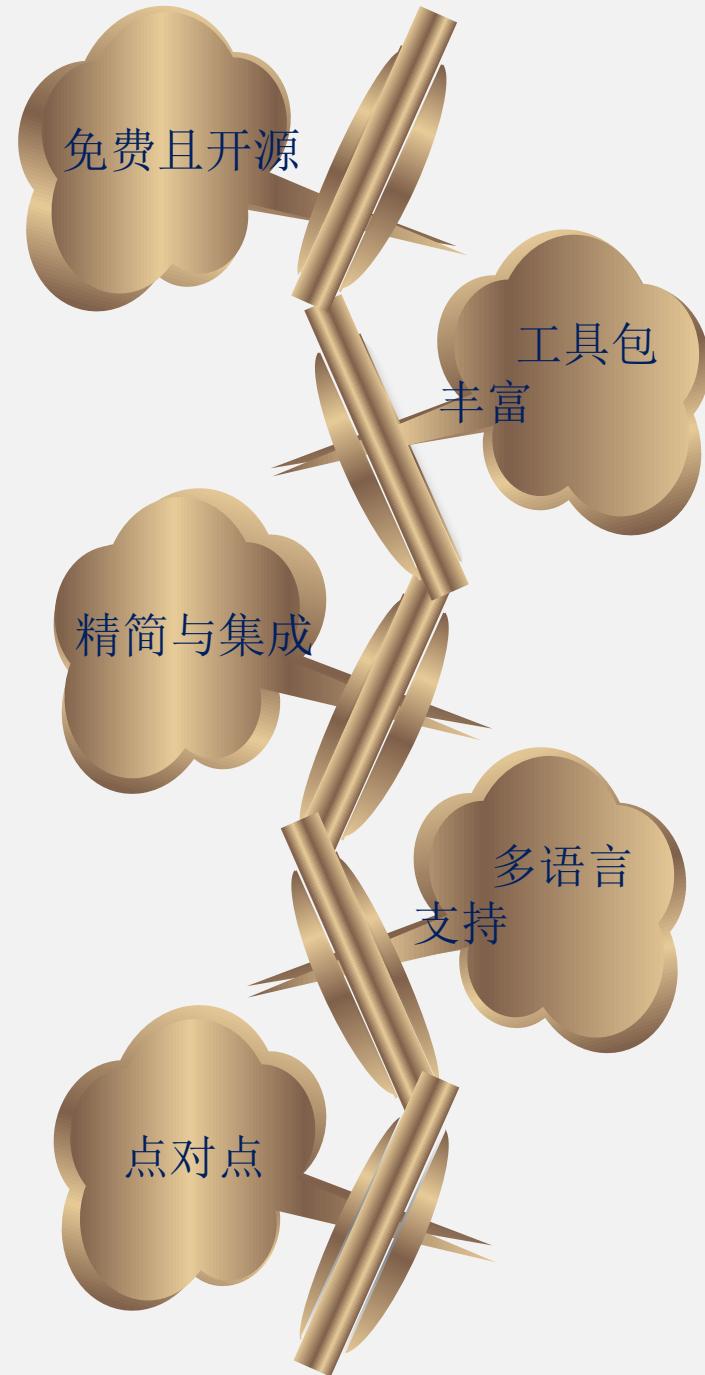
2，丰富的机器人功能库方便快速搭建原型

3，便利的数据记录、分析、仿真工具，方便调试

4，学界和产业界的 standard 方便学习和交流

- 使用ROS，能够方便迅速地搭建好机器人原型。

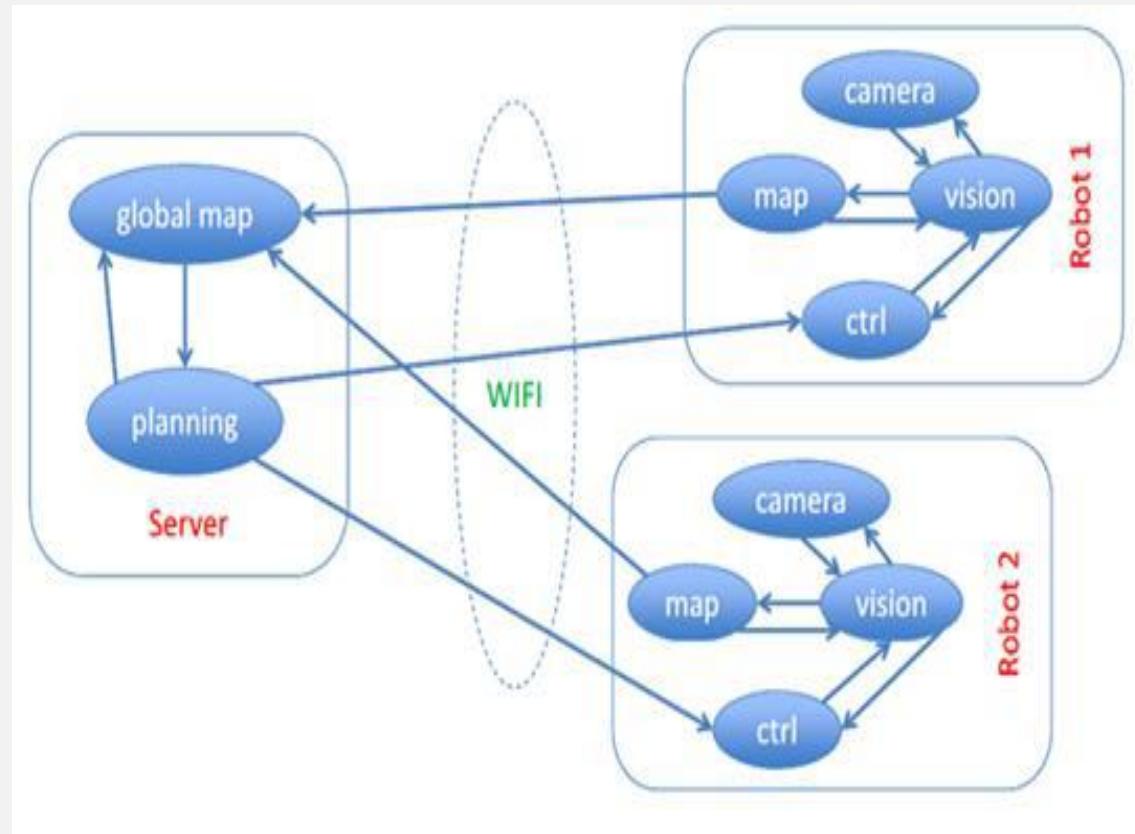
ROS总体设计 —五个特点



ROS的点对点设计

一个使用ROS的系统包括一系列进程，这些进程存在于多个不同的主机并且在运行过程中通过端对端的拓扑结构进行联系。虽然基于中心服务器的那些软件框架也可以实现多进程和多主机的优势，但是在这些框架中，当各电脑通过不同的网络进行连接时，中心数据服务器就会发生问题。

ROS的点对点设计以及服务和节点管理器等机制可以分散由计算机视觉和语音识别等功能带来的实时计算压力，能够适应多机器人遇到的挑战。



ROS的多语言支持

ROS现在支持许多种不同的语言，例如C++、**Python**、**Octave**和**LISP**，也包含其他语言的多种接口实现。



```
→ ~ rosmsg show geometry_msgs/Twist
geometry_msgs/Vector3 linear
  float64 x
  float64 y
  float64 z
geometry_msgs/Vector3 angular
  float64 x
  float64 y
  float64 z
```

运动控制接口

```
→ ~ rosmsg show sensor_msgs/Image
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
  uint32 height
  uint32 width
  string encoding
  uint8 is_bigendian
  uint32 step
  uint8[] data
```

图像接口

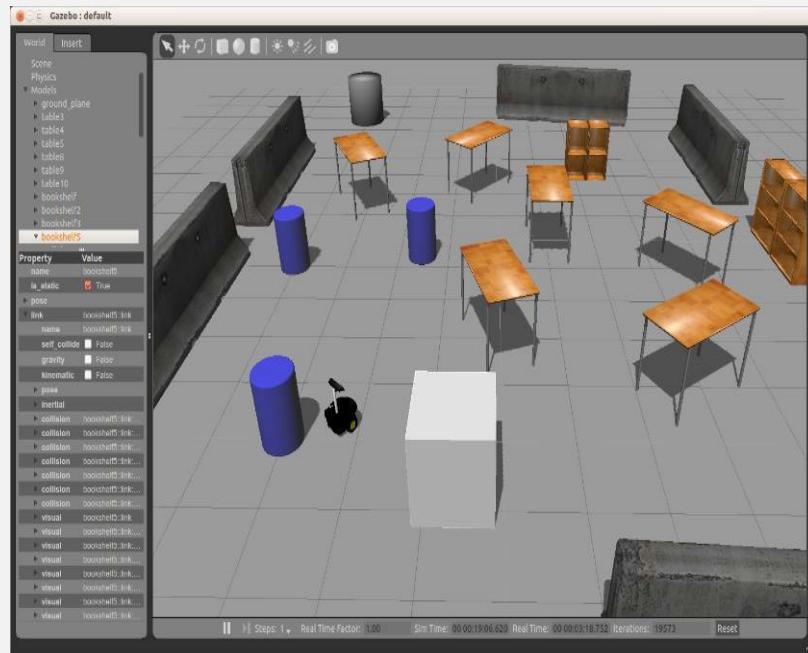
ROS架构精简，集成度高

- 每个功能节点单独编译；
- 集成众多跨平台开源项目；
- 同一接口，代码复用率高。

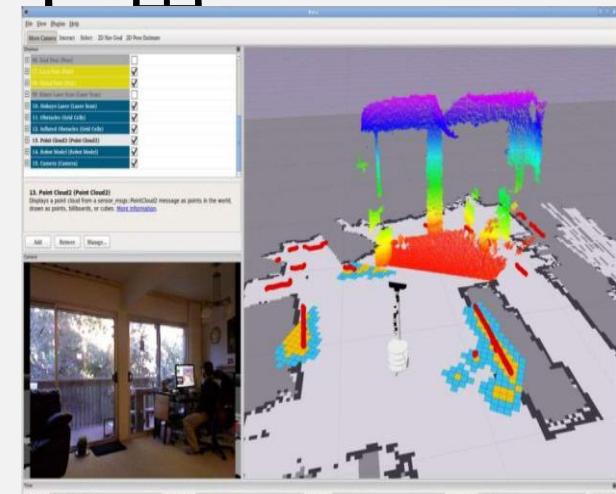


ROS的组件化，工具包丰富

- 3D可视化工具rviz;
- 物理仿真环境gazebo;
- 数据记录工具 rosbag;
- QT工具箱 rqt*



Gazebo



Rviz



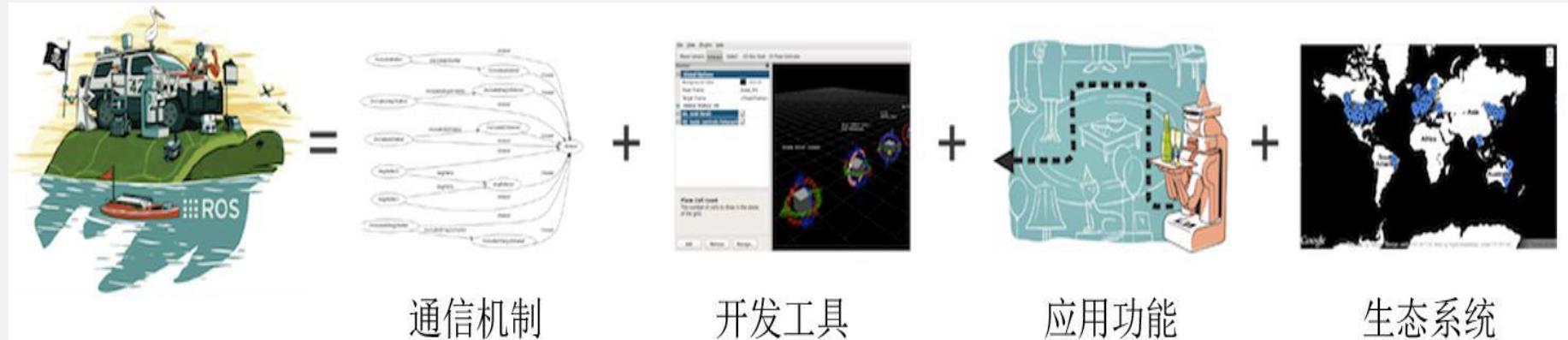
QT工具箱

ROS免费且开源

- BSD许可，可修改，可复用，可商用；
- 软件包数量级数增长，良好生态环境。



ROS的四位一体



ROS提供了全新的通信机制，全面的开发工具，多样的应用功能，完整的生态系统。提供了机器人开发的统一标准。

ROS Robots



<http://wiki.ros.org/cn/ROS/Introduction>

ROS

Robots



KUKA



iRobot



Google OMRON

SoftBank

Microsoft

iRobot®

BostonDynamics

国际知名企业机器人企业以ROS作为机器人研发首选



NVIDIA. 大疆創新



BOSCH

QUALCOMM



Nvidia、博世、高通、英特尔、宝马、大疆等

国内外各大公司纷纷推出ROS接口

最庞大的使用者群体，事实上的机器人标准

- 2016年ROS大会数据：目前在使用操作系统做开发的人员用户超过35万。（个人用户中国第二）

1.	 United States	85,023 (23.06%)
2.	 China	62,933 (17.07%)
3.	 Japan	30,745 (8.34%)
4.	 Germany	28,022 (7.60%)
5.	 India	12,918 (3.50%)

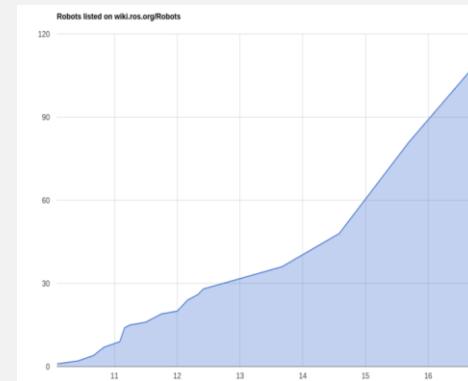
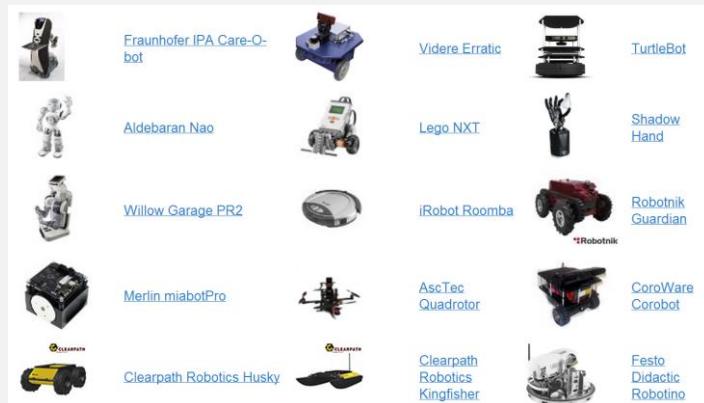


学术研究领域： 目前学术领域大部分最新成果都会发布ROS的版本

- “ROS: an open source Robot Operating System” (Quigley et al., 2009) :
- 论文引用量**2871**
- Source: Google Scholar 2017-01-03

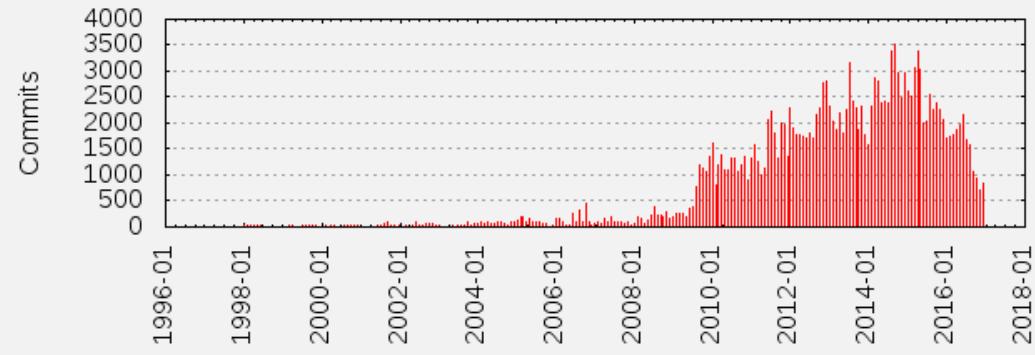
2016年官方支持的ROS机器人100+

- <http://wiki.ros.org/Robots>
- 产品：目前已经有很多机器人公司采用了ROS系统来开发一些应用于全新市场的产品，如ClearPath, Rethink, Unbounded, Neurala, Blue River, Big-i, 最典型的就是Willow Garage的PR2机器人。
- 投资机构的热捧：仅2015年，相关风险投资机构就在基于ROS操作系统的机器人公司投资了超过1.5亿美元。
- 大公司的支持：Nvidia、博世、高通、英特尔、宝马以及大疆等。



代码统计

- The total line count is over **14 million** lines of code
- There have been **2477** authors
- And 181509 commits
- Averaging 73.3 commits per author
- 代码语言
 - cpp: 2608592 (63.98%)
 - python: 553332 (13.57%)
 - ansic: 297629 (7.30%)
 - xml: 280615 (6.88%)
 - lisp: 149439 (3.67%)
 - java: 135343 (3.32%)
 - ruby: 26484 (0.65%)
 - sh: 21120 (0.52%)



对于开发者而言：快速搭建机器人原型

➤ 通过ROS这个平台学习先进机器人技术

掌握机器人框架体系构建

学习先进的机器人算法

跟踪当前最先进的硬件技术

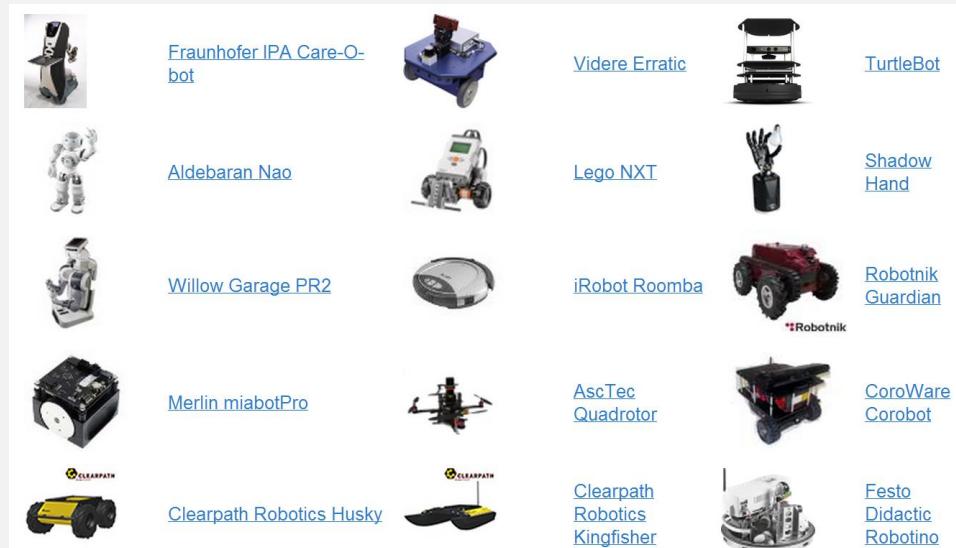
把握技术潮流跟踪技术前沿

ROS的不足

- 目前基于Ubuntu系统，移植嵌入式系统困难
- 无实时性设计
- 体积较大
- 系统整体运行效率低
- 总的来说，ROS不擅长开发成熟的商业产品。

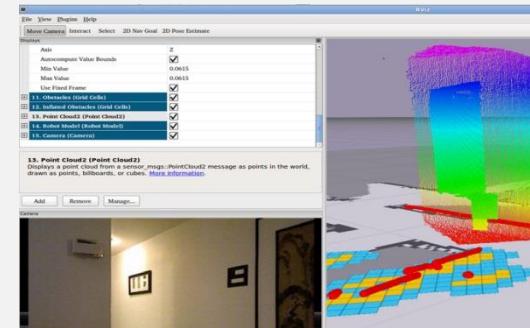
ROS应用场景（1）机器人

- Mobile manipulator : PR2
- Flight control system : pixhawk
- Multiple sensor: hokuyo sick pointgrey
- 更多: <http://wiki.ros.org/Robots>



ROS应用场景（2）先进算法及行业应用

- ros-industrial



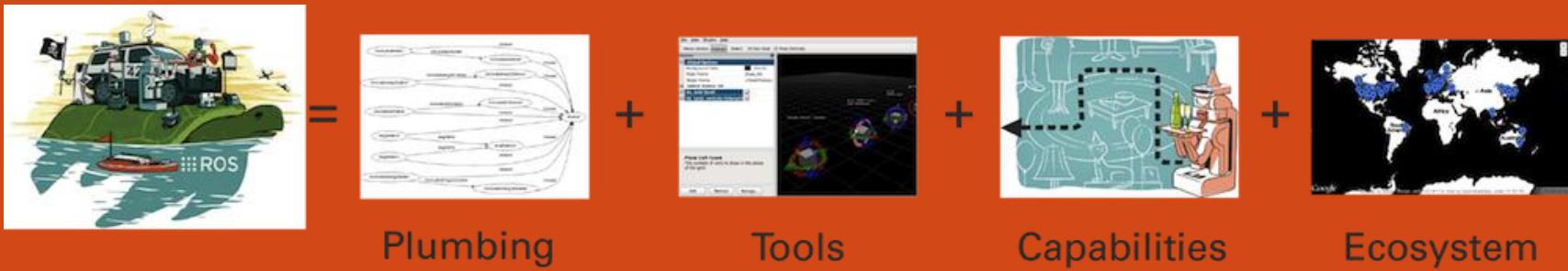
- state_of_art algorithm: lsd_slam svo ompl etc....
- Software Architecture : ros_control moveit



ROS应用场景：总结

- 商业上的机器人产品一般会给出ROS的软件接口（比如ROS-1基于的工业机器人的MoveIt接口）商业上很多机器人相关软件也会给出ROS接口（比如webots仿真器， matlab的robot system toolbox），从一方面说明ROS的普及程度。
- ROS现在最广泛的用途：算法与硬件的快速结合，系统集成方案快速构建、评估与验证。
- 现阶段不建议将ROS**整体、直接**用于商业产品。
- ROS 2.0：新的架构（取消了Master），新的通信机制（DDS数据分发协议），对实时性的支持，有望实现嵌入式应用。

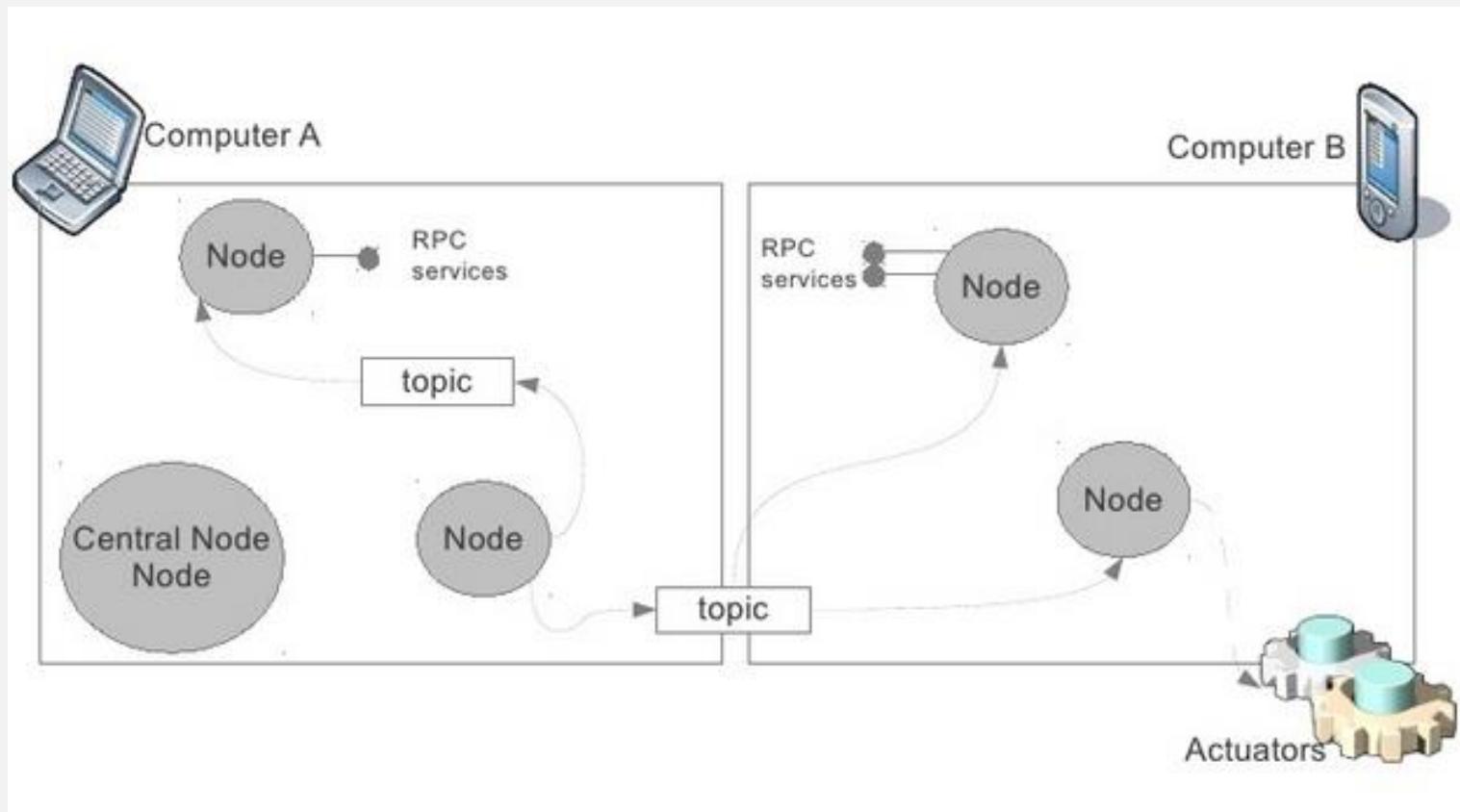
What to Learn



Contents

- 1、认识ROS
- 2、ROS初级命令
- 3、ROS中级命令
- 4、机器人系统设计
- 5、机器人仿真
- 6、机器人感知与交互
- 7、机器人SLAM与自主导航
- 8、机械臂控制MoveIt！
- 9、机器人系统综合开发

设计思想：分布式架构



ROS核心概念

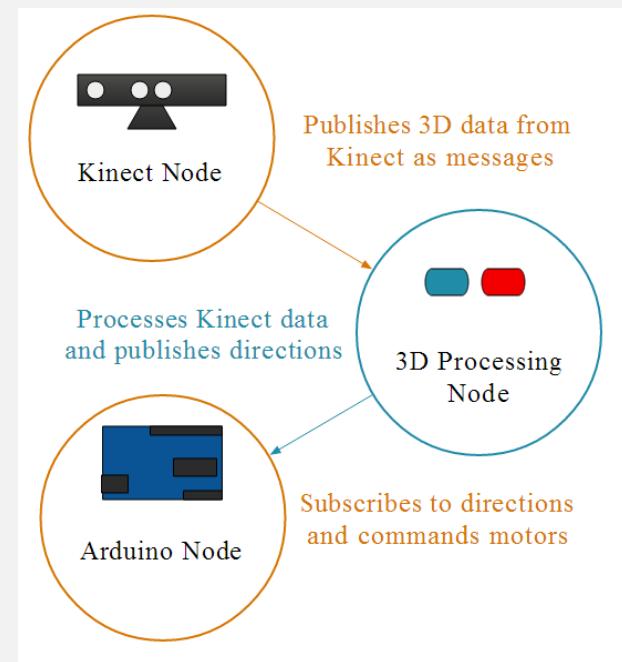
- Nodes 节点
- Messages and Topics 消息与话题（或主题）
- Services 服务
- ROS Master ROS管理器
- Parameters 参数
- Stacks and packages 功能包集与功能包

ROS Nodes

- 节点是各自独立的可执行文件，能够通过话题、服务或参数服务器与其他进程（节点）通信。
- ROS通过使用节点将代码和功能解耦，提高了系统容错能力和可维护性，使系统简化。同时，节点允许ROS系统布置在任意多个机器上并同时运行。
- 节点在系统中必须有唯一的名称。
- 节点可以使用不同的库进行编写，如roscpp和 rospy。
 - roscpp基于C++
 - rospy基于Python。

ROS Topics

- 话题是节点间用来传输数据的总线。
- 1-to-N Publish/Subscribe 模式：同一个话题也可以有很多个订阅者。
- 使用 TCP/IP 传输，称为 TCPROS，ROS 默认。
- 使用 UDP 传输，称为 UDPROS，适合于远程操控任务。（低延迟高效率的传输方式，但可能产生数据丢失）。



ROS Messages

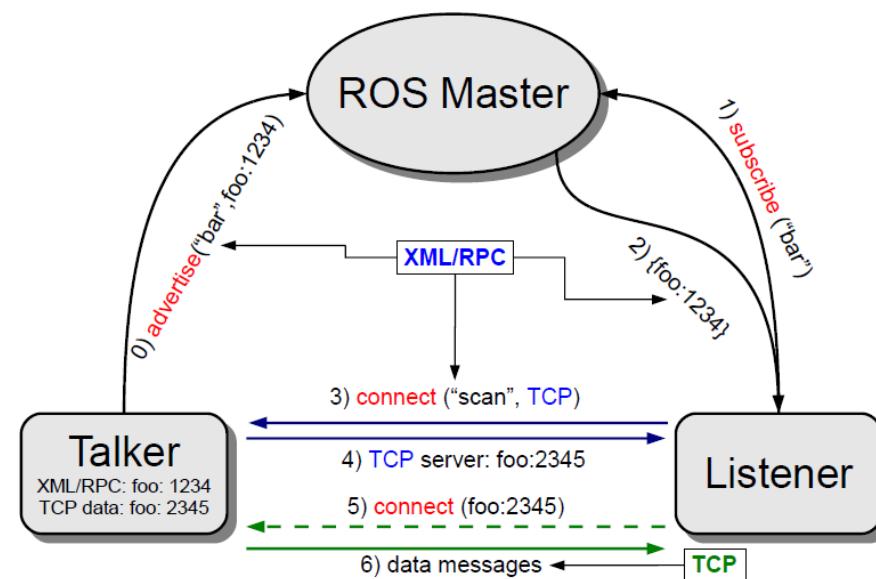
- 一个节点通过向特定话题发布消息。
- 消息具有一定的类型和数据结构，包括ROS提供的标准类型和用户自定义类型。
- 消息的类型标准命名方式进行约定：功能包名称/.msg文件名称。
- 最常用的有
 - geometry_msgs/Twist 包括线速度和角速度
 - Vectors三维向量

ROS Services

- **1-to-1Service/Client** 模型：当直接与节点通信并获得应答时，将无法通过话题实现，这时需要使用服务。
- 服务需要由用户开发，节点并不提供标准服务。
- 命令行工具：
 - rossrv查看有关服务数据结构的信息
 - rosservice列出服务列表和查询某个服务

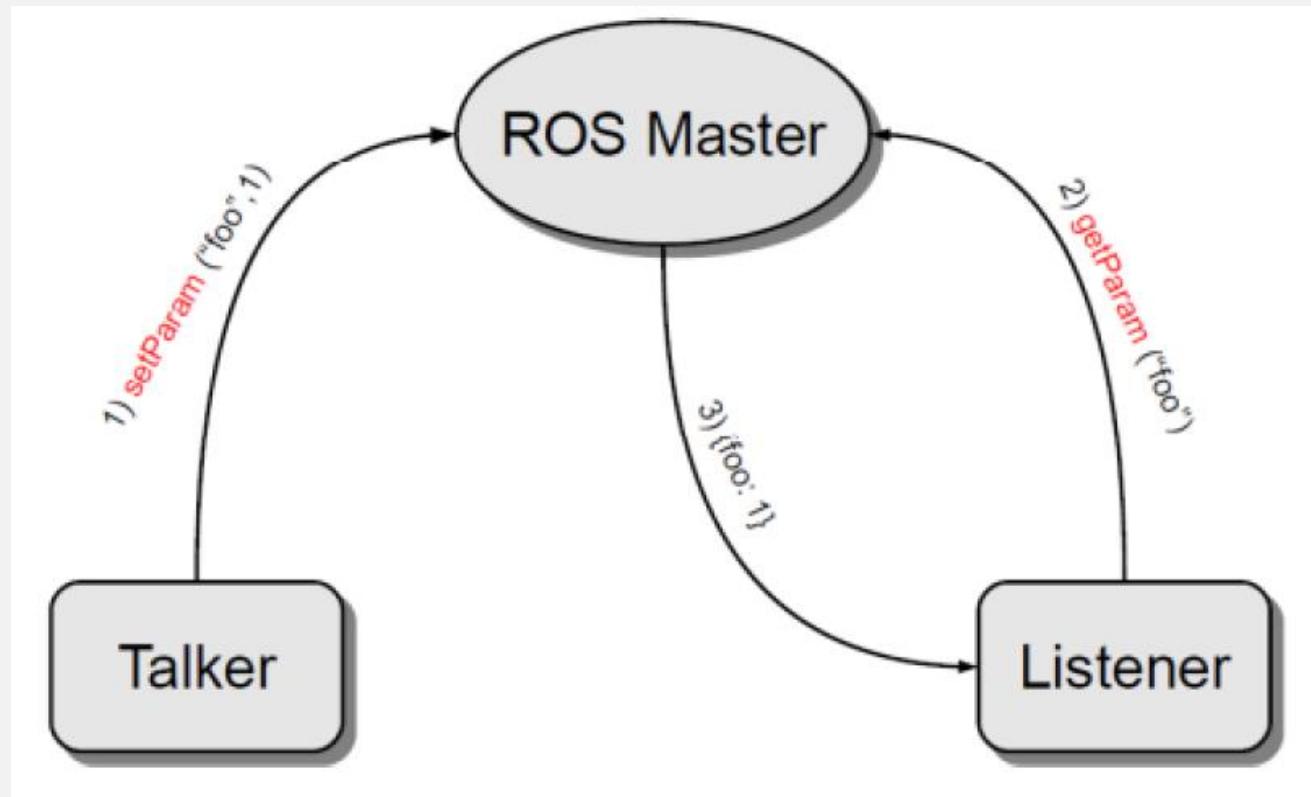
ROS Master

- 向ROS系统中其他节点提供命名和注册服务
- 跟踪和记录话题的发布者和订阅者
 - 使ROS节点之间能够相互查找。一旦节点找到了彼此，就能建立一种点对点的通信方式。
- 提供参数服务器



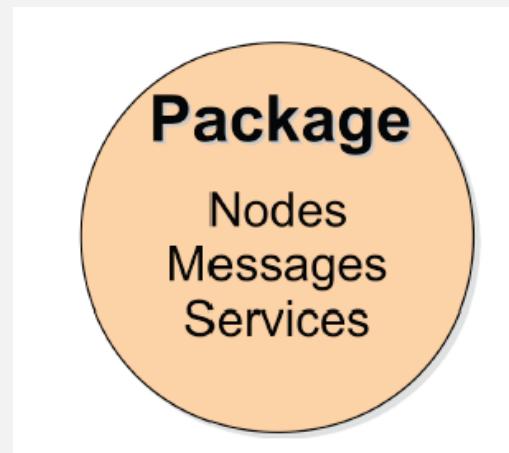
Parameter Server

- 参数服务器是可通过网络访问的共享的多变量字典。
- 节点使用此服务器来存储和检索运行时的参数。

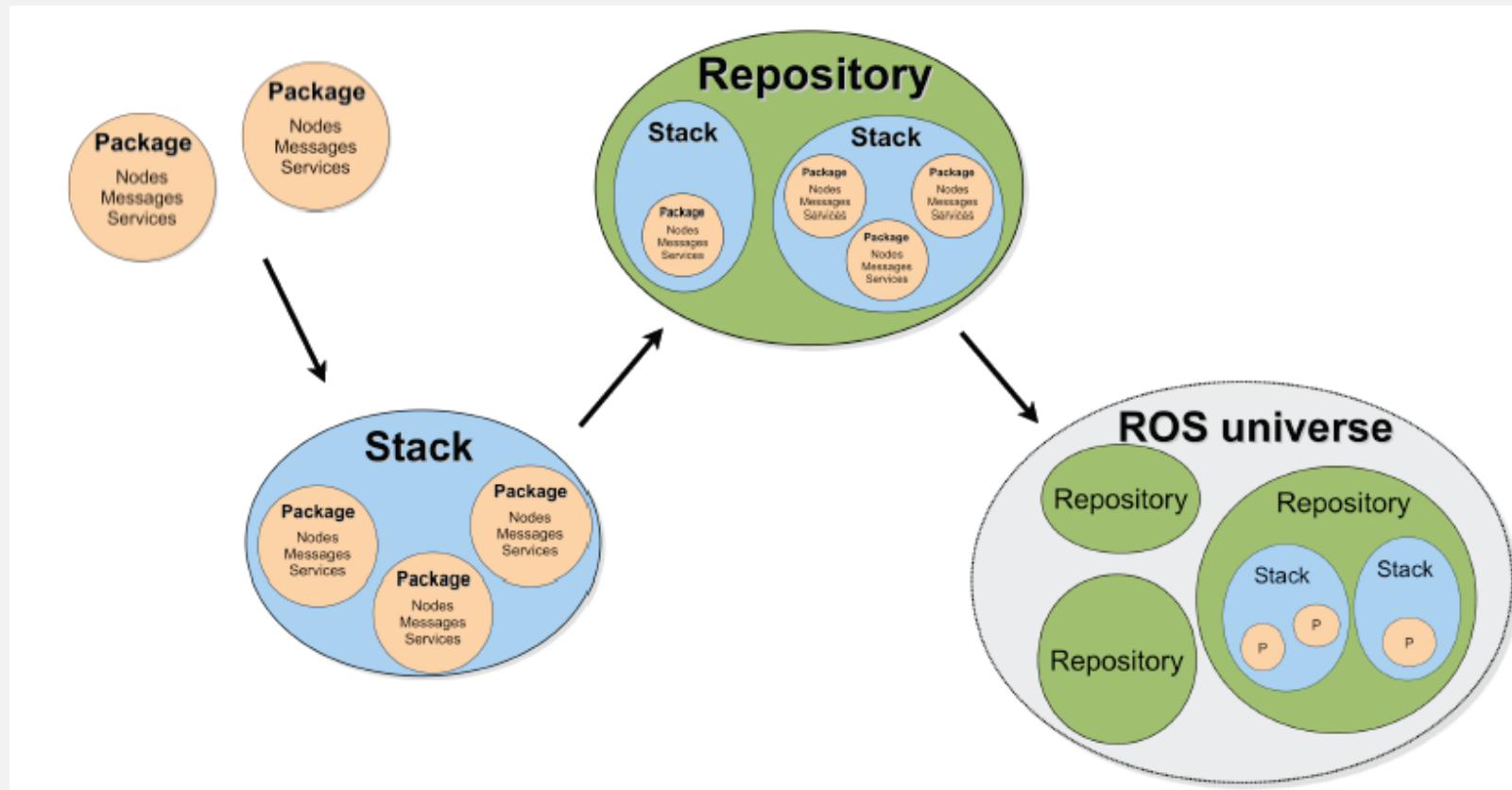


ROS Packages

- 功能包（Package） ROS中软件组织的基本形式，用于创建ROS程序。包含ROS运行的进程（节点）、配置文件等。
- 功能包清单（Manifest） 功能包清单提供关于功能包、许可信息、依赖关系、编译标志等的信息。功能包清单是一个manifests.xml文件，通过这个文件能够实现对功能包的管理。

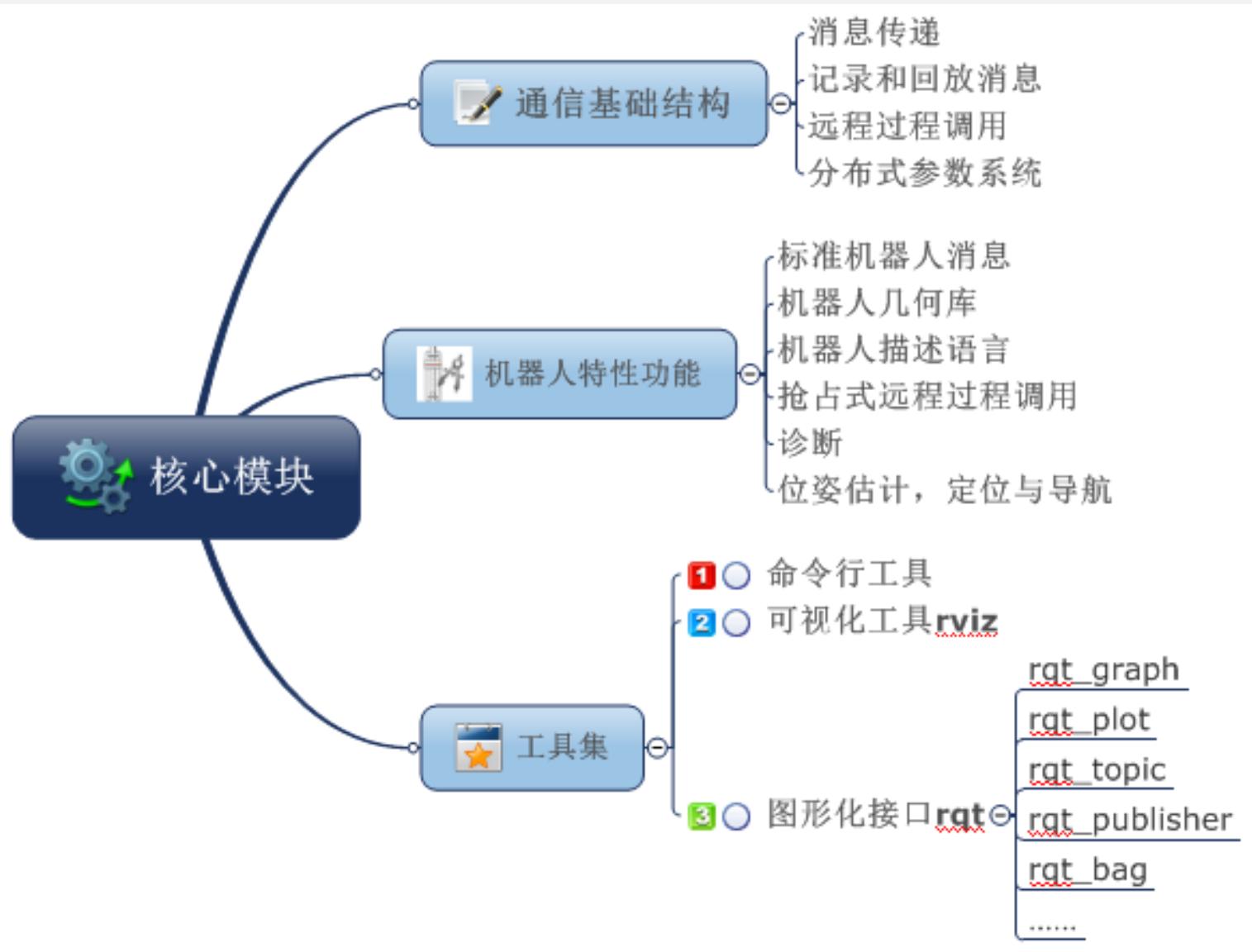


ROS Package System



Taken from Sachin Chitta and Radu Rusu (Willow Garage)

核心模块

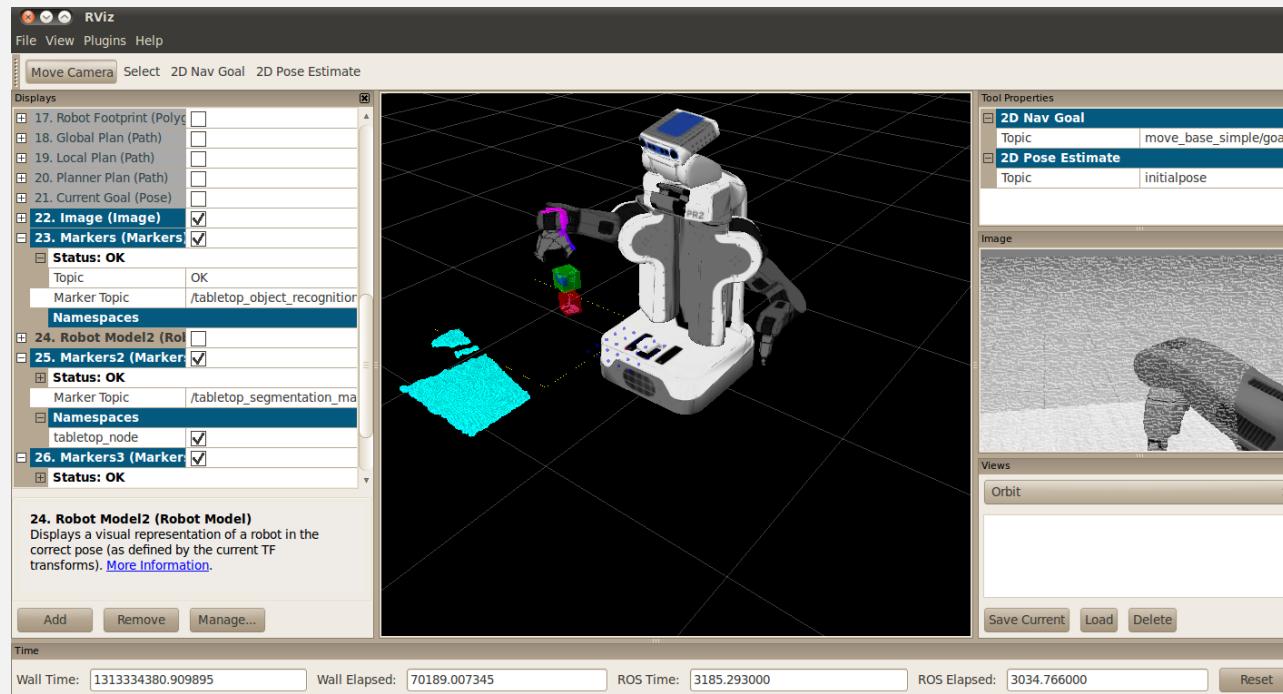


ROS 常用命令工具

- rostopic (Topics)
- rosservice (Services)
- rosnode (Nodes)
- rosparam (Parameters)
- rosmsg (Messages)
- rossrv (Services)
- rosrtf (General debugging)

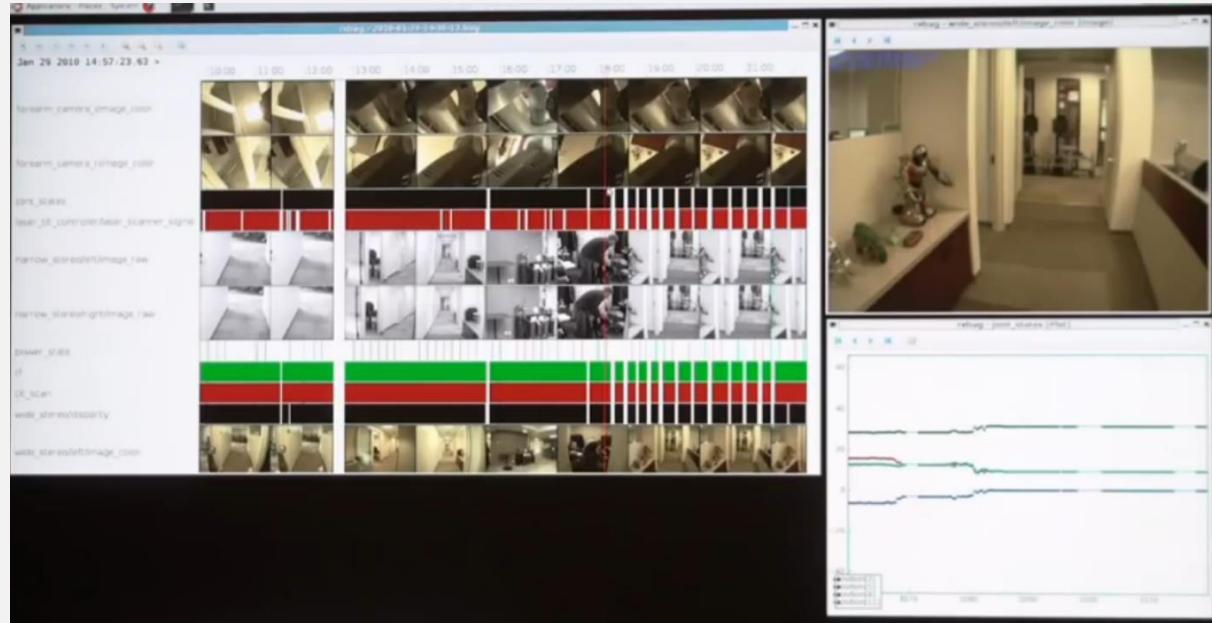
ROS 可视化工具

- 便于机器人仿真与观察
- rqt – 集成图像交互界面
- rviz – 3D 可视化工具



ROS 数据存储/回放

- 思想：使用bag存储topic（例如现实中的传感器数据），以后调用bag的topic数据则不必每次都在现实中运行机器人。
- Rosbag：高性能，直接生成topic，避免数据转换。



ROS log系统

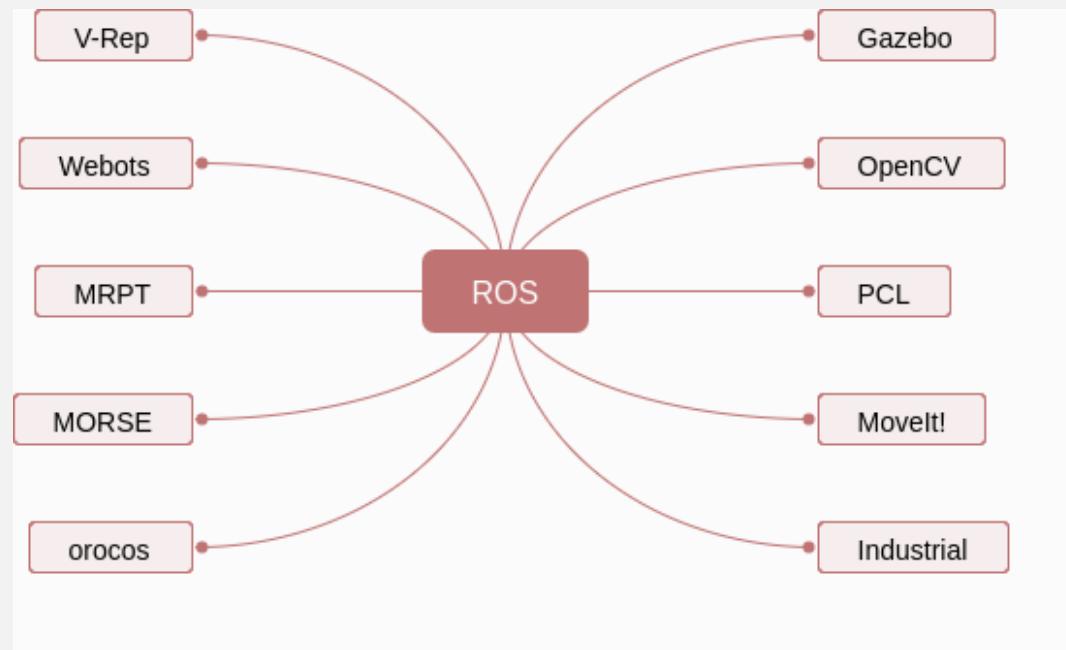
- 记录软件运行相关信息
- 基于log4cxx
- rosconsole： 提供多个层级的log 信息

The screenshot shows the 'rqt_console' application window titled 'rqt_console__Console - rqt'. The window has a toolbar with icons for file operations and a status message 'Displaying 30 messages'. Below the toolbar is a table with columns: #, Message, Severity, Node, Stamp, Topics, and Location. The table lists several log entries, primarily from the '/move_base' node, showing various info and warn messages related to odometry and base recovery.

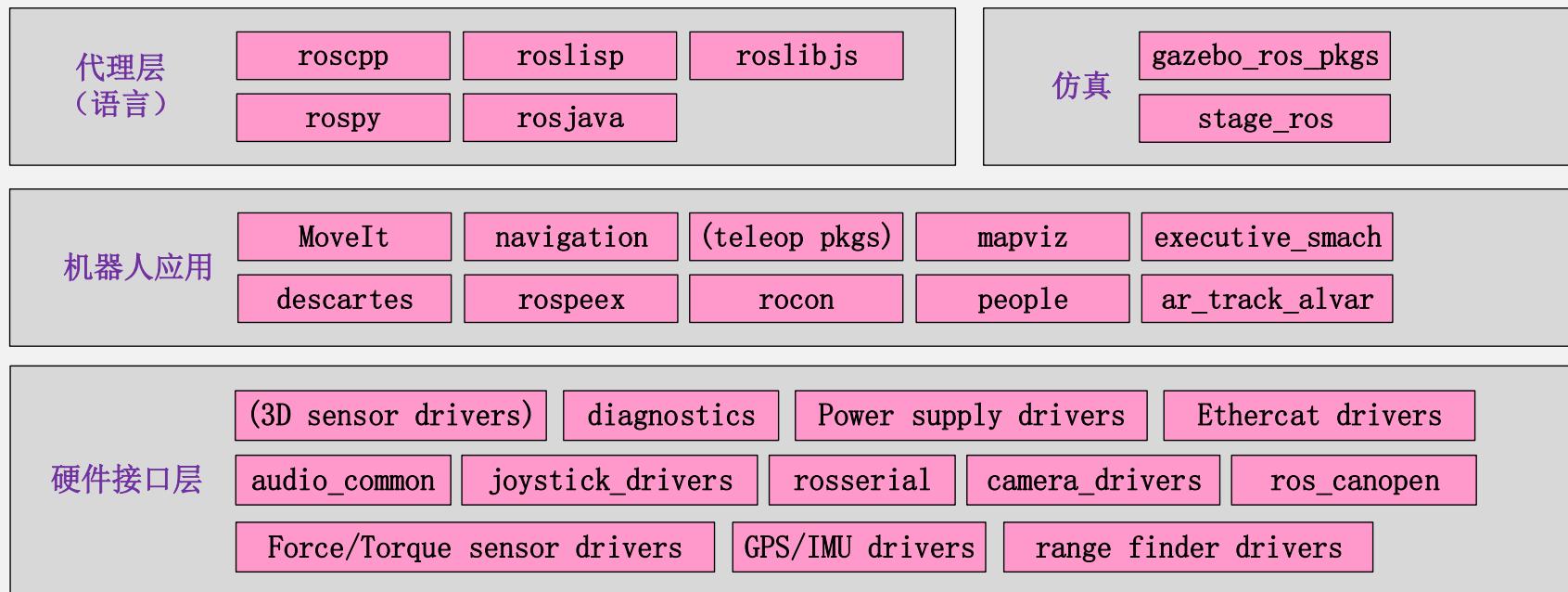
#	Message	Severity	Node	Stamp	Topics	Location
#30	i odom re...	Info	/move_bas...	16:00:03.20...	/cmd_vel, /...	/tmp/build...
#29	! The bas...	Warn	/move_bas...	16:00:03.20...	/cmd_vel, /...	/tmp/build...
#28	i Recover...	Info	/move_bas...	16:00:03.10...	/cmd_vel, /...	/tmp/build...
#27	! The bas...	Warn	/move_bas...	16:00:03.10...	/cmd_vel, /...	/tmp/build...
#26	i Recover...	Info	/move_bas...	16:00:03.00...	/cmd_vel, /...	/tmp/build...
#25	! The bas...	Warn	/move_bas...	16:00:03.00...	/cmd_vel, /...	/tmp/build...

进一步

- 结合硬件平台和具体应用学习使用各种package
 - OpenCV、PCL、Moveit!、Control、Navigation、SLAM
- 在调试过程中学习使用调试仿真工具
 - Rqt、rviz、Gazebo
- 开发测试新的机器人算法
- 集成各类package构建机器人原型

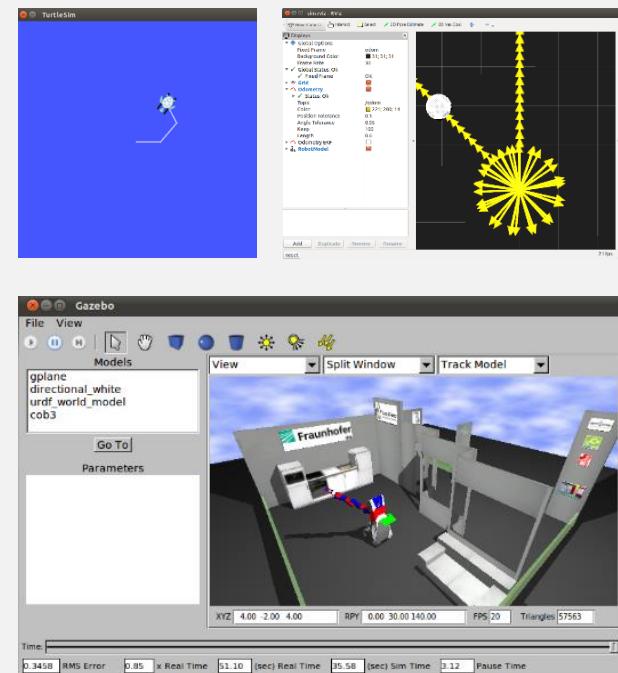


学习ROS的Package



仿真环境：由易至难

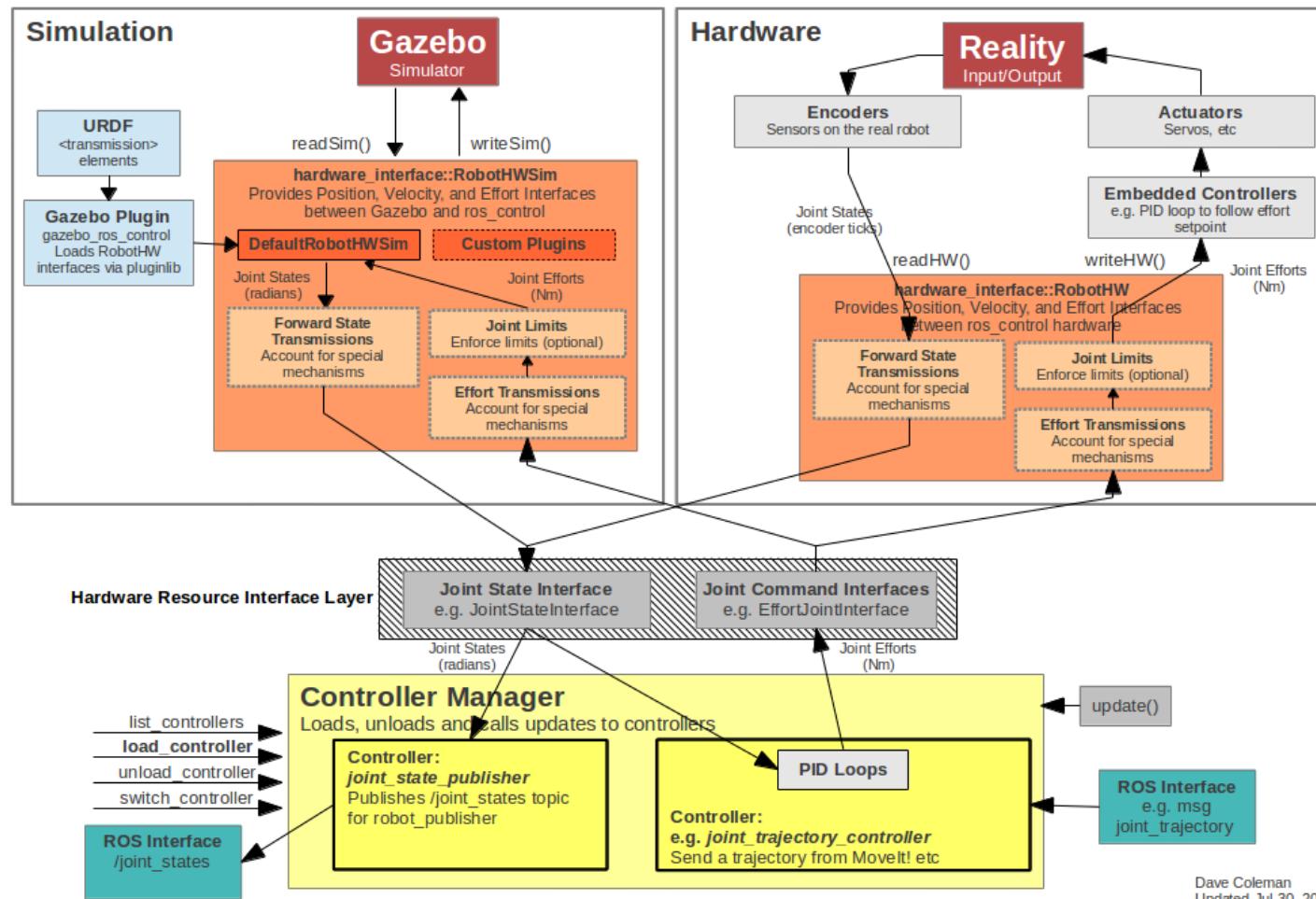
- Turtlesim
 - 一个QT开发的2D轨迹显示界面
 - 只能显示运动轨迹
- ArbotiX
 - 含有一个差速驱动机器人的rviz模拟器
 - 机器人运动及topic数据的3D显示
- Gazebo
 - 功能齐全的3D物理模拟器
 - 非常重，对内存和显卡要求高



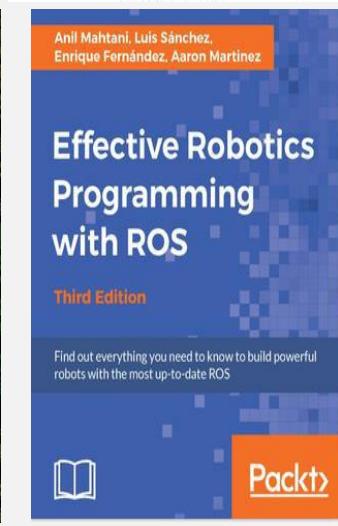
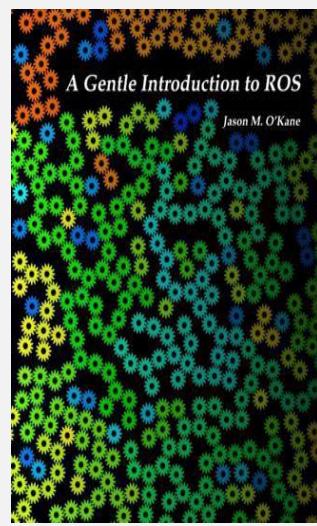
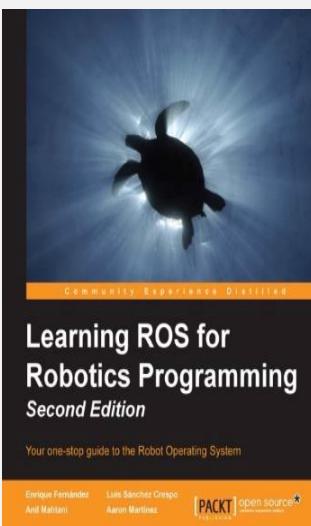
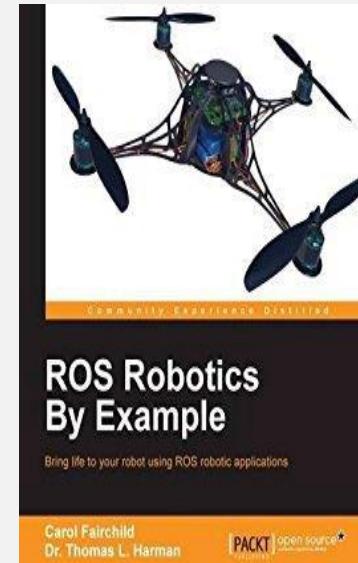
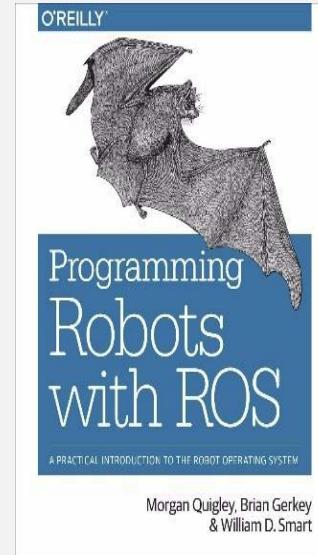
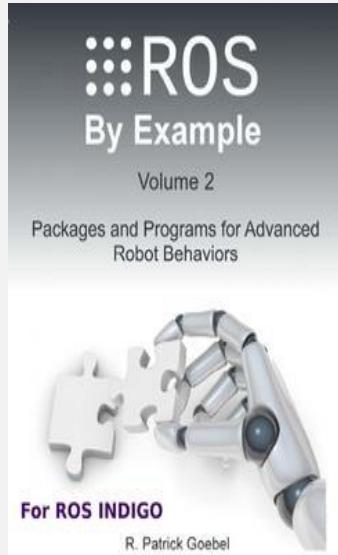
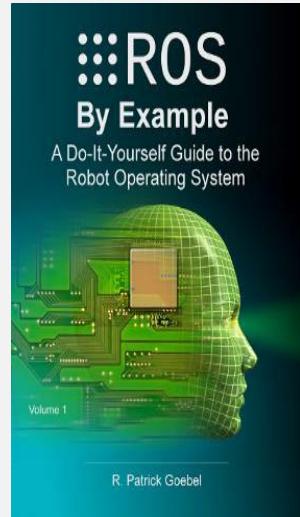
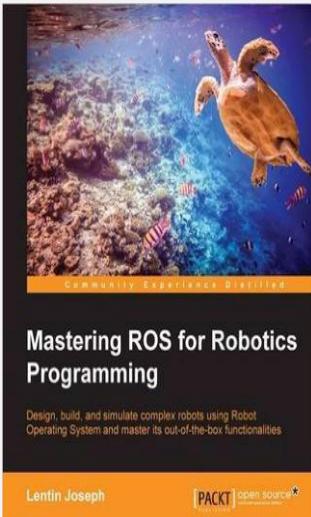
Gazebo 控制仿真



GAZEBO + ROS + ros_control



参考书目



Enjoy!