



# 机器人原理与应用

Principles and Applications of Robotics

**Instructor:** Dr. 阎华松(Huasong Min), professor

**Office Location:** 钢铁楼1110(Room No. 1110, GANGTIE Building)

**Class venue:** Room No. F4201

**Email:** [mhuasong@wust.edu.cn](mailto:mhuasong@wust.edu.cn)

**Mobile:** 13971365898

# CONTENTS

The course is divided into eight modules covering the following areas:

-  **绪论**  
Introduction and Conceptual Problems
-  **机器人系统分析基础**  
System Model of Robot
-  **运动学**  
Robot Kinematics
-  **动力学**  
Robot Dynamics
-  **机器人运动规划**  
Robot Motion Planning
-  **机器人控制**  
Robot Control
-  **机器人编程语言**  
Programming Language of Robot
-  **典型机器人系统的设计与实现**  
Design and Implementation of Robot System

This is an introductory of robotics course, containing both fundamental as well as some more advanced concepts. It presents a broad overview of robotics with focus on manipulators and mobile robots, and includes robot kinematics, dynamics, planning and control, programming language.

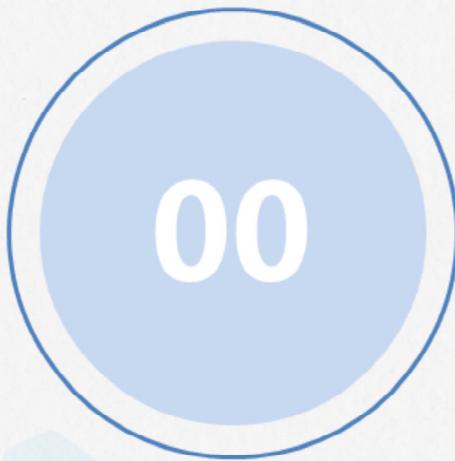
The course is divided between the following areas:

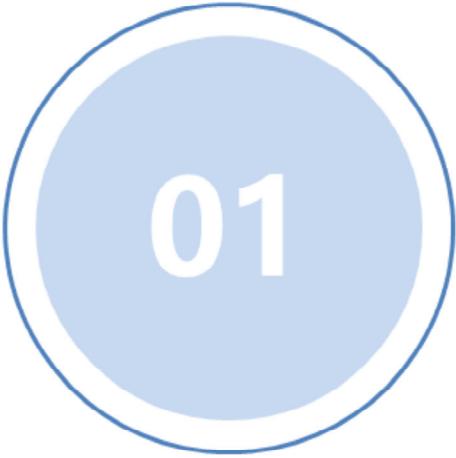
- Robotics Introduction
- System Model of Robot
- Robot kinematics
- Robot dynamics
- Robot control
- Robot motion planning

...

## 机器人系统分析与设计基础

- 机器人与人工智能(Robotics and AI)
- 智能机器人体系结构(Architecture of Intelligent Robots)
- 机器人系统的组成(System of Robots)
- 机器人位形(Configure of Robot and C-Space)
- 机器人关节类型与自由度(Type and Freedom of Robot Joint)
- 常用的机器人建模描述文件与仿真平台(Robot Description and Simulation)
- 机器人系统的设计方法(Design Method of Robot System)
- 基于ROS的机器人系统设计 (Robot System Design on ROS)
- 扩展阅读(Expand Knowledge)





01

## 机器人与人工智能

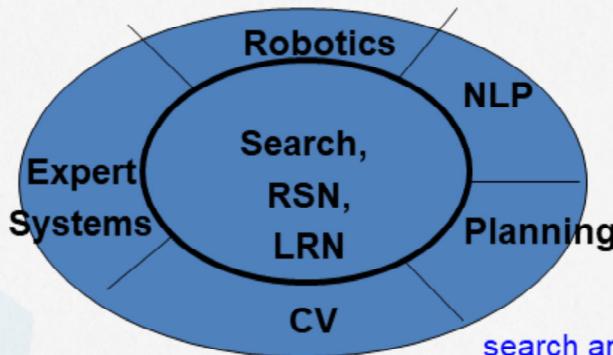
Robotics and AI

# 机器人与人工智能

Robot and AI

## Symbolism, Connectionism and Behaviorism

- A perspective of AI
- Artificial Intelligence - Knowledge based computing
- Disciplines which form the core of AI - inner circle
- Fields which draw from these disciplines - outer circle.
- Symbolism, Connectionism and Behaviorism?



- ✓ AI源自1955年的达特茅斯会议
- ✓ 逻辑学派（符号主义）
  - 物理符号系统假设和有限合理性原理
- ✓ 仿生学派（连接主义）
  - 人工神经网络和进化计算
- ✓ 控制论学派（行为主义）
  - 感知和行为之间的关系

search and mathematical optimization, logic, methods based on probability and economics

there are at least three kinds of AI, such as Symbolism, Connectionism and Behaviorism.  
This aspect of classification may be obsolete because modern algorithms usually have those two or all property.

符号主义、连接主义、行为主义？

search and mathematical optimization, logic, methods based on probability and economics

LR logic Reasoning  
logical deduction  
mathematical optimization

# 机器人与人工智能

Robot and AI

## Symbolic AI 符号主义

### ☆ Symbolic AI - Physical Symbol System Hypothesis

- Every intelligent system can be constructed by storing and processing symbols and nothing more is necessary.

符号主义(symbolicism)，又称为逻辑主义(logicism)、心理学派(psychologism)或计算机学派(computerism)，其原理主要为物理符号系统(即符号操作系统)假设和有限合理性原理。

### ☆ Symbolic AI has a bearing on models of computation such as

- Turing Machine
- Von Neumann Machine
- Lambda calculus

传统的人工智能认为人工智能源于数理逻辑，主张以知识为基础，通过推理来进行问题求解，在研究方法上采用计算机模拟人类认知系统功能的功能模拟方法

Simon、Minsky和Newell等认为，人和计算机都是一个物理符号系统，因此可用计算机的符号演算来模拟人的认知过程；作为智能基础的知识是可用符号表示的一种信息形式，因此人工智能的核心问题是知识表示、知识推理和知识运用的信息处理过程。

(1) 符号主义(symbolicism)，又称为逻辑主义(logicism)、心理学派(psychologism)或计算机学派(computerism)，其原理主要为物理符号系统(即符号操作系统)假设和有限合理性原理。

**Symbolic** artificial intelligence is the term for the collection of all methods in artificial intelligence research that are based on high-level "symbolic" (human-readable) representations of problems, logic and search. **Symbolic AI** was the dominant paradigm of AI research from the mid-1950s until the late 1980s.

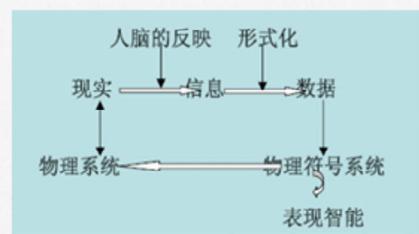
● 即传统的人工智能，认为人工智能源于数理逻辑，主张以知识为基础，通过推理来进行问题求解，在研究方法上采用计算机模拟人类认知系统功能的功能模拟方法

● Simon、Minsky和Newell等认为，人和计算机都是一个物理符号系统，因此可用计算机的符号演算来模拟人的认知过程；作为智能基础的知识是可用符号表示的一种信息形式，因此人工智能的核心问题是知识表示、知识推理和知识运用的信息处理过程。

# 机器人与人工智能

Robot and AI

## Challenges to Symbolic AI



### ☆ Motivation for challenging Symbolic AI

☆ A large number of computations and information process tasks that living beings are comfortable with, are not performed well by computers!

### 脑计算

- ◆ Brain computation in living beings
- Pattern Recognition
- Learning oriented
- Distributed & parallel processing
- Content addressable

### The Differences

### 图灵计算

- TM computation in computers
- Numerical Processing
- Programming oriented
- Centralized & serial processing
- Location addressable

TM = Turing Machines

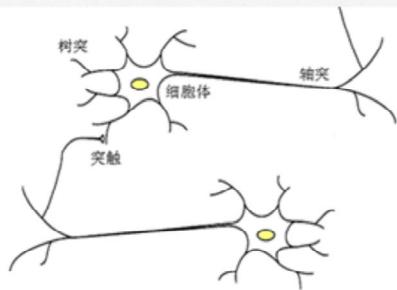
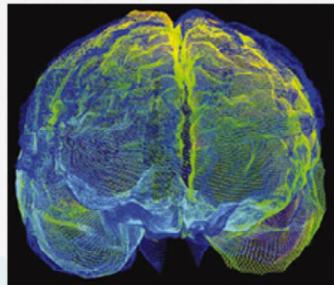
脑计算, 图灵计算

# 机器人与人工智能

Robot and AI

## Connectionist AI 连接主义

脑神经网络 -> 计算机神经网络



### Artificial Neural Networks

#### 人工神经网络

- 树突是树状的神经纤维接收网络，它将电信号传送到细胞体
- 细胞体对这些输入信号进行整合并进行阈值处理
- 轴突是单根长纤维，它把细胞体的输出信号导向其他神经元
- 一个神经细胞的轴突和另一个神经细胞树突的结合点称为突触
- 神经元的排列和突触的强度(由复杂的化学过程决定)确立了神经网络的功能。

连接主义(Connectionism)，又称为仿生学派(Bionicsm)或生理学派(Physiologism)，其主要原理为神经网络及神经网络间的连接机制与学习算法。

• 大脑含约 $10^{11}$ 个神经元，它们通过 $10^{15}$ 个联结构成一个网络。每个神经元具有独立的接受、处理和传递电化学信号的能力，这种传递由神经通道来完成

• 树突是树状的神经纤维接收网络，它将电信号传送到细胞体

• 细胞体对这些输入信号进行整合并进行阈值处理

• 轴突是单根长纤维，它把细胞体的输出信号导向其他神经元

• 一个神经细胞的轴突和另一个神经细胞树突的结合点称为突触

• 神经元的排列和突触的强度(由复杂的化学过程决定)确立了神经网络的功能。

(2) 连接主义(connectionism)，又称为仿生学派(bionicsm)或生理学派(physiologism)，其主要原理为神经网络及神经网络间的连接机制与学习算法。

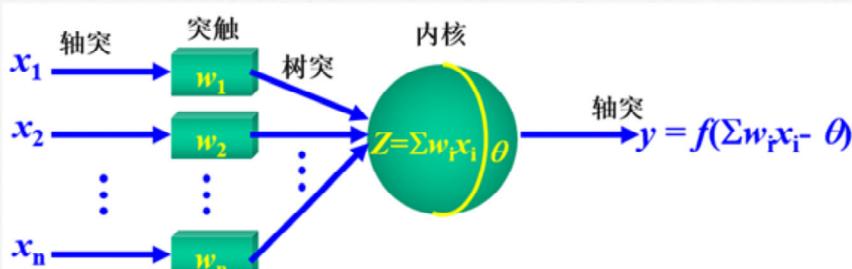
Connectionism is a set of approaches in the fields of artificial intelligence, cognitive science and philosophy of mind, that attempts to represent mental or behavioral phenomena as emergent processes of interconnected networks of simple units. There are many forms of connectionism, but the most common forms use neural network models.

人脑神经网络 -> 计算机神经网络

# 机器人与人工智能

Robot and AI

## Connectionist AI 连接主义



神经元获得网络输入信号后，信号累计效果整合  
函数 $u(x)$ 大于某阈值 $\theta$ 时，神经元处于激发状态；  
反之，神经元处于抑制状态。

适合大规模并行处理、容错性、  
自组织和自适应能力和联想功  
能强



### 神经网络从四个方面刻画人脑的基本特征

Warren McCulloch

Walter Pitts

- **物理结构**: 模仿生物神经元的功能，构造人工神经元的联结网络
- **计算模拟**: ANN以具有局部计算能力的神经元为基础，同样实现信息的大规模并行处理。
- **存储与操作**: 大脑对信息的记忆是通过改变突触的强度来实现并分布存储。ANN模拟信息的大规模分布存储。
- **训练**: 后天的训练使得人脑具有很强的自组织和自适应性。ANN根据人工神经元网络的结构特性，使用不同的训练过程，自动从“实践”（即训练样本）中获取相关知识，并存储在系统中。

神经网络从四个方面刻画人脑的基本特征：

物理结构：模仿生物神经元的功能，构造人工神经元的联结网络

计算模拟：ANN以具有局部计算能力的神经元为基础，同样实现信息的大规模并行处理。

存储与操作：大脑对信息的记忆是通过改变突触的强度来实现并分布存储。ANN模拟信息的大规模分布存储。

训练：后天的训练使得人脑具有很强的自组织和自适应性。ANN根据人工神经元网络的结构特性，使用不同的训练过程，自动从“实践”（即训练样本）中获取相关知识，并存储在系统中。

The central connectionist principle is that mental phenomena can be described by interconnected networks of simple and often uniform units. The form of the connections and the units can vary from model to model. For example, units in the network could represent neurons and the connections could represent synapses like in the brain of a human being.

# 机器人与人工智能

Robot and AI

## 行为主义(Behaviorism or Behaviourism)

- 认为人工智能来源于控制论，智能取决于感知和行动。提出智能行为的“感知—动作”模式，采用行为模拟方法  
a systematic approach to understanding the behavior of humans and other animals
- 对符号主义、联结主义采取批判的态度；（智能不需要知识、表示和推理，只需要与环境交互作用）  
Most behavior-based systems are also reactive, which means they need no programming of internal representations of what a chair looks like, or what kind of surface the robot is moving on. Instead all the information is gleaned from the input of the robot's sensors. The robot uses that information to gradually correct its actions according to the changes in immediate environment.
- 20世纪80年代诞生智能控制和智能机器人系统学科（R. A. Brooks），为机器人研究开创了新的方法。Behavior-Based Robots (BBR)

在人工智能研究领域，有关“符号主义”、“行为主义”、“联结主义”的争论一直都还没有停息。“符号主义”可以得到有限已知条件下的确定解，但对于复杂的系统难以用物理符号表述；“行为主义”可以得到未知环境下的快速反应，但不能明确表示它要完成的目标和为达到目标需要实施的行动，难以设计直接目标的行为；这两种形式的混合结构可以兼顾长期目标和快速反应的需求，达到一种折中的策略。

(3) 行为主义(actionism)，又称为进化主义(evolutionism)或控制论学派(cyberneticsm)，其原理为控制论及感知-动作型控制系统。

Behaviorism (or behaviourism) is a systematic approach to understanding the behavior of humans and other animals. It assumes that all behaviors are either reflexes produced by a response to certain stimuli in the environment, or a consequence of that individual's history, including especially reinforcement and punishment, together with the individual's current motivational state and controlling stimuli. Although behaviorists generally accept the important role of inheritance in determining behavior, they focus primarily on environmental factors.

Behavior-based robotics or behavioral robotics is an approach in robotics that focuses on robots that are able to exhibit complex-appearing behaviors despite little internal variable state to model its immediate environment, mostly gradually correcting its actions via sensory-motor links.

Behavior-based robotics sets itself apart from traditional artificial intelligence by using biological systems as a model. Classic artificial intelligence typically uses a set of steps to solve problems, it follows a path based on internal representations of events compared to the behavior-based approach. Rather than use preset calculations to tackle a situation, behavior-based robotics relies on adaptability. This advancement has allowed behavior-based robotics to become commonplace in researching and data gathering.

Most behavior-based systems are also reactive, which means they need no programming of internal representations of what a chair looks like, or what kind of surface the robot is moving on. Instead all the information is gleaned from the input of

the robot's sensors. The robot uses that information to gradually correct its actions according to the changes in immediate environment.

Behavior-based robots (BBR) usually show more biological-appearing actions than their computing-intensive counterparts, which are very deliberate in their actions. A BBR often makes mistakes, repeats actions, and appears confused, but can also show the anthropomorphic quality of tenacity. Comparisons between BBRs and insects are frequent because of these actions. BBRs are sometimes considered examples of weak artificial intelligence, although some have claimed they are models of all intelligence.

在人工智能研究领域，有关“符号主义”、“行为主义”、“联结主义”的争论一直都还没有停息。“符号主义”可以得到有限已知条件下的确定解，但对于复杂的系统难以用物理符号表述；“行为主义”可以得到未知环境下的快速反应，但不能明确表示它要完成的目标和为达到目标需要实施的行动，难以设计直接目标的行为；这两种形式的混合结构可以兼顾长期目标和快速反应的需求，达到一种折中的策略。

# 机器人与人工智能

Robot and AI

## AI的发展

能理解会思考

03

能听会说、能看会认

02

能存会算

01

第一阶段

计算智能

第二阶段  
感知智能

第三阶段  
认知智能

计算智能-能存会算

感知智能-能听会说、能看会认

认知智能-能理解会思考

人工智能的研究经历了计算智能、感知智能的发展过程，目前正朝认知智能的阶段发展，机器人智能水平伴随人工智能的研究也在不断提高，特别是在2016年3月，AlphaGo打败了世界围棋冠军李世石，说明了机器人与人工智能的研究将迎来又一个春天，走上又一个高度。有关神经网络，包括深度学习、强化学习等先进算法在内的人工智能算法及架构，将广泛应用于智能机器人的总体设计中。

# 机器人与人工智能

Robot and AI

Learning by  
observations and  
explanations



Data-driven learning

感知智能  
以模式识别、浅层理解为特征  
以大数据和云计算为支撑

Learning by doing



Trial-and-error learning

认知智能  
以深层语义理解、知识推理为特征  
新一代的类人智能关键算法

NLP  
HRI  
Reasoning  
Machine Learning  
LLM

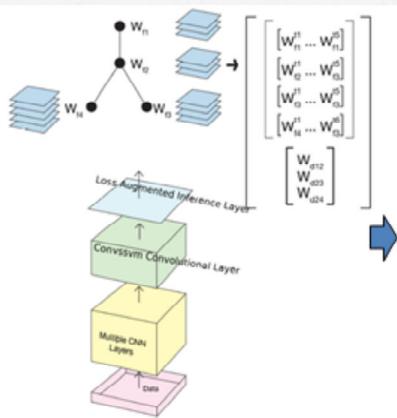


# 机器人与人工智能

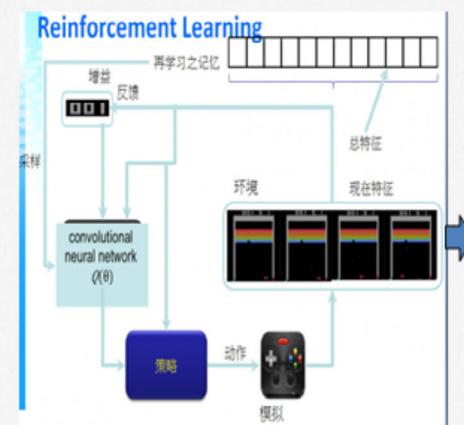
Robot and AI

## 机器学习 Machine Learning

### 深度学习 Deep Learning



### 强化学习 Reinforcement Learning



### 迁移学习 Transfer Learning



机器人学习 Robot Learning: 模仿学习 Imitation Learning or LfD(Learn from Demo)



02

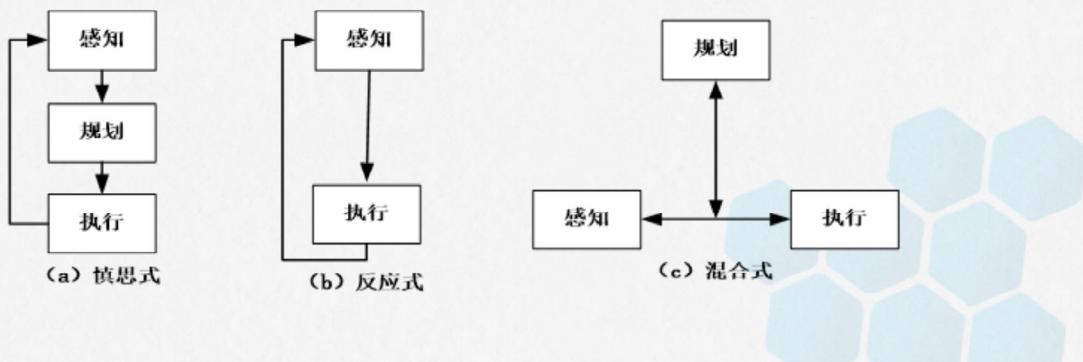
## 机器人软件体系结构

机器人体系结构是定义机器人系统功能模块以及模块之间逻辑关系的结构框架，是实现整个机器人系统的基础。体系结构既是机器人本体的物理框架，又是机器人智能的逻辑载体。从机器人体系结构上来对机器人进行定义，对于机器人及其系统的设计具有更高的指导意义。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 智能机器人软件体系结构三种范式



机器人除了人工智能的软件实现技术外，还涉及机构、控制、通信等方面的技术。从机器人硬件组成的角度出发，由于机器人的形态各异，很难形成一个统一、标准的硬件体系结构。从人工智能的角度出发，对机器人软件体系结构进行设计，可以在更高的层面对系统设计进行统一的指导。

一般来说，如果从机器人软件控制、人工智能的角度出发，我们可以将智能机器人体系结构分为慎思式、反应式和混合式三种基本范式

所谓的慎思式结构（Deliberative Architecture），基于物理符号系统假说，系统的决策通过逻辑推理、模式匹配以及符号操作来完成，它比较适合已知环境下任务明确、目标给定的系统，将系统认知功能模块化（如感知、学习、规划和动作），可以大大降低系统的复杂性，但其存在的最大问题是由于现实世界的复杂性和不确定性，符号推理的应用范围非常局限，一旦改变环境，符号推理必须重新修改，灵活性不足，而且由于其动作并不是传感器数据直接作用下的结果，而是经历感知、建模到规划等处理阶段之后的结果，需要大量的计算，实时性不好。

反应式结构（Reactive Architecture）是基于行为范例的智能体体系结构，它直接根据传感器的输入进行决策，在其内部不维护环境模型，没有世界模型和规划器，根据行为模式，以刺激—回答的方式来对环境的改变做出反应，低层次的行为反应和高层次的行为反应具有不同的优先级。基于行为的反应模式具有较好的实时性，响应快、灵活，可以在不同环境下，针对输入的传感信号，做出相应的反应。在这种模式下，智能的实现是基于“感知-动作”模式，但如果要实现更高的智能，比如按照人类的智能一样逐步进化（又被称为进化模式），这种模式不能明确表示它要完成的目标和为达到目标需要实施的行动，难以设计直接目标的行为。

混合式结构，则是以上两种方式的结合，具有两个或两个以上的模块，慎思式模块采用符号表示和符号推理，反应式模块，直接对环境做出反应，系统

既体现规划推理，又实时反馈交互，它根据输入，先经过反应式模块，如果是快速反应，则直接根据行为模式输出控制，如果需要推理，则把输入传入慎思式模块进行推理产生决策。这种结构兼顾了长期目标和快速反应的需求，是一种折中的实现方式。

上面这三种体系结构基本是基于人工智能的智能体(Agent)模式进行的分类。在机器学习中，机器人的分类方法有很多种，比如按机器人的几何结构、控制方式、信息输入方式、智能程度、用途、移动性等来进行分类。机器人除了人工智能的软件实现技术外，还涉及机构、控制、通信等方面的技术。从机器人硬件组成的角度出发，由于机器人的形态各异，很难形成一个统一、标准的硬件体系结构。从人工智能的角度出发，对机器人软件体系结构进行设计，可以在更高的层面对系统设计进行统一的指导。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 分类角度?

- ✓ 几何结构、控制方式、信息输入方式、智能程度、用途、移动性?
- ✓ 机构、控制、通信、软件实现技术?
- ✓ 综合人工智能的agent模式、控制系统的方式、通信、仿生学、脑科学的模型
  - 层次结构
  - 包容结构
  - 分布式结构
  - 仿生结构

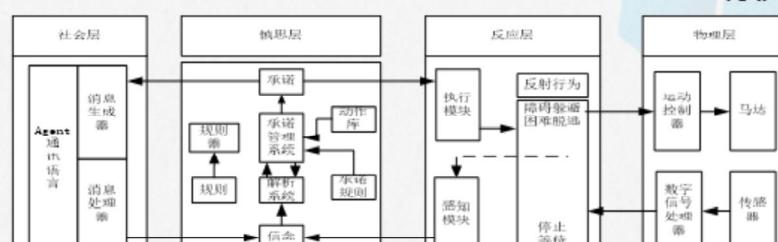
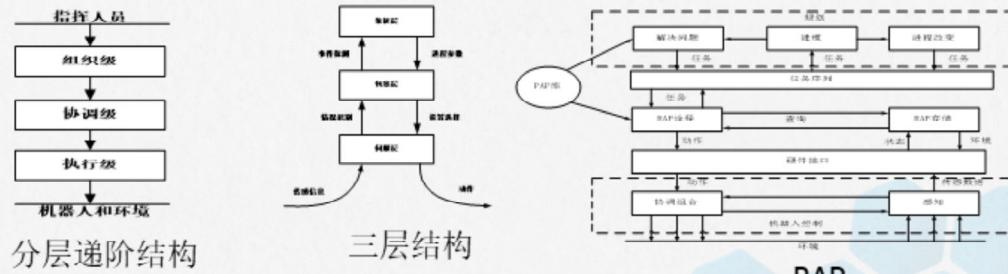


有关机器人软件体系结构的研究一直处于发展的过程中，综合人工智能的智能体(Agent)模式、控制系统的方式、仿生学、脑科学的模型，可以归纳出如下几种机器人软件体系结构：层次结构、包容结构、分布式结构、仿生结构等类型，根据机器人系统的需求，可采用不同形式设计满足要求的机器人系统。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 层次结构 (Hierarchical Architecture)



层次结构是将各功能模块分成若干层次，使不同层次上的模块具有相互独立性，同时高级别的层次可以通过输出信息来控制低级别的层次，低级别层次的功能可以支持高层功能实现。

层次结构的早期代表为Saridis在1979年提出的智能控制系统分层递阶结构，如图2.2所示。其分层原则是：随着控制精度的增加，智能能力减弱，即层次向上智能增加，但是精度降低。他根据这一原则把智能控制系统分为三级，即组织级、协调级和执行级。分层递阶结构通过信息的自顶向下逐层传递，间接地控制行为，其智能部分分布在顶层。因为该结构是通过自顶向下任务逐层分解，问题求解精度逐层增加，从而实现了从抽象到具体、从人工智能推理方法到具体动作控制的过渡。该结构的优点是具有很好的推理能力，较好地解决了智能与控制精度的关系。其缺点是信息是自顶向下单向传输，系统的可靠性、鲁棒性、反应性不足。

另一种典型的层次结构是三层结构。IBM的T.J. Watson Research Center于1992年提出了一种包含象征层（Symbolic layer）、包容层（Subsumption Layer）以及伺服层（Servo-control Layer）的三层结构，如图2.3所示。加州理工学院喷气推进实验室于1998年提出了一种由反馈控制层、序列层和慎思层组成的三层结构。其信息流程也是从低层传感器开始，经过内外状态的形势评估、归纳，逐层向上，且在高层进行总体决策；高层接受总体任务，根据信息系统提供的信息进行规划，确定总体策略，形成宏观命令，再经协调级的规划设计，形成若干子命令和工作序列，分配给控制器加以执行。这些层次结构很好地保证了智能机器人在时间上对环境的准确把握，但是该类结构忽视了传感信息的融合、学习。

第三种有代表性的层次结构是RAP(Reactive Action Packages)结构。这种层次结构大致由规划、执行和控制三部分组成，其结构如图2.4所示。

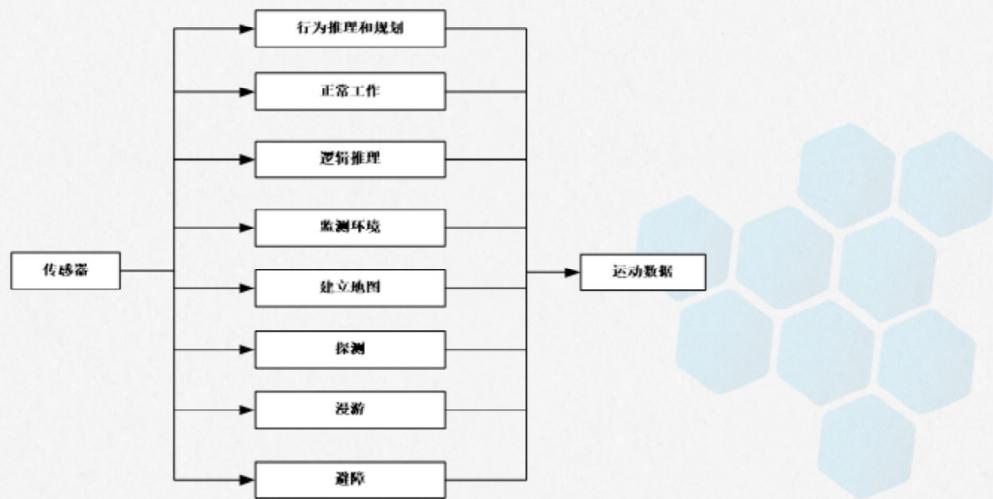
RAP结构是耶鲁大学的Robert James Firby在1989年提出来的智能机器人软件体系结构，该结构使用规划器将机器人的任务粗略的分解成任务序列，然后执行这些任务序列以完成目标。执行系统执行规划器提供的任务，并把这些任务进行细化，最后控制系统将这些任务翻译成低级的命令给执行器完成动作。当系统的任务执行失败的时候，系统将尝试候补的方法来完成任务，直到所有的方法都失败了，任务才会被系统放弃，这样使系统具有模块化、鲁棒性和柔性等特征。

在1999年，为了完成多机器人的控制协调，Rooney等人根据社会智能假说提出了由物理层、反应层、慎思层和社会层构成的社会机器人软件体系结构（Social Robot Architecture），如图2.5所示。社会机器人结构是基于三层结构发展而来的，它把原有的三层结构重新细化，并加入了多传感融合系统，同时为了智能体间的交互加入了社会层。物理层指机器人本体，主要包含电机和传感器等部件；反应层定义了一系列的基本动作，当慎思层传来动作命令时将会把动作解析为相应的动作序列，并转化为控制信号传给电机，同时也把传感信息转化为智能体事件传给慎思层。慎思层负责对智能体的意识属性进行维护，依据环境状态的变化和动作执行的结果及时调整自身的信念和愿望，以激发做出新的承诺生成新的意图。社会层负责与其他智能体进行协调、协作，并通过对自身和外界的共同意图进行融合，形成智能体消息传给慎思层。其特色在于BDI（信念Belief、愿望Desire、意图Intention）模型的慎思层和基于智能体通讯语言Teanga的社会层，BDI模型赋予了机器人心智状态，Teanga赋予了机器人社会交互能力。社会机器人结构采用社会智能对机器人建模，能很好的描述智能机器人的智能、行为、信息、控制的时空分布模式，引入智能体理论可以对机器人的智能本质进行更细致的刻画，对机器人的社会特性进行更好的封装。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 包容体系结构 (Subsumption Architecture)



通过对移动机器人的计算需求进行分析，Brooks等人提出了一种基于行为分解的控制结构，从行为角度将控制问题分解为一系列并行排列的“层”，如图2.6所示。这种分解方式将控制系统分解成一组能单独对机器人进行控制的子系统，每一个子系统都能完成特定的功能。在最初的研究中，Brooks将一个机器人系统从下到上共分为8层，标号为0-7：1)避障(Avoid Objects)：障碍物可以是动态的或者静止的；2)无碰撞的随机漫游(Wander)；3)探测(Explore)；4)建立环境模型(Build Maps)，对目的地进行路径规划；5)监测“静态”环境中的动态变化(Monitor Changes)；6)对目标任务进行逻辑推理(Identify Objects)；7)通过适当的方法执行任务(Plan to Change the World)；8)行为推理和规划(Reason about Behavior of Objects)。

包容结构将系统的控制分布等一系列异步或并行处理的行为层中，每一个行为层只关注问题的一个方面。包容结构的所有信息流都是单向的从传感器到执行器，层级代表了该层行为的控制优先级。层次间存在密切联系，通过“硬”控制的方法，上层行为可以包含下层行为，对下层行为的输出产生抑制作用。

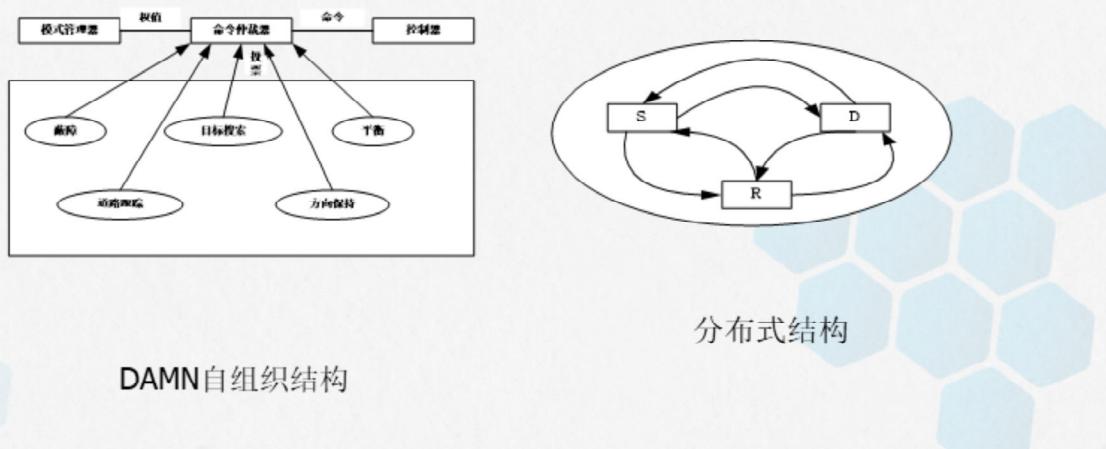
Brooks认为，对一个机器人控制系统来说，它必须能同时满足多个任务，甚至是多个相互冲突的任务要求。有关这一点，在包容结构中是通过级别高的命令完全覆盖级别低的控制命令来实现的。执行机构不需要知道是哪一层控制机器人，而且主导机器人运动的控制层也不需知道被包容层的信息，这在很大程度上降低了机器人完成复杂任务的难度。但其也有弊端，具体体现在：1)在选择控制命令时，为了做出一个最合适的决定，需要权衡许多因素。有时，一种级别高的行为抑制另一种级别低的行为是不恰当的，它可能完全丢

失被抑制层的信息。例如，第3层决定控制机器人，那么第0层就不能抵抗，而第1层有避障的功能，第3层却没有。为了解决这一问题，就要求每一层必须能执行所有比它低的层的功能。这就使得更高行为层的设计变得越来越复杂。**2)包容结构**将行为封装在一系列有限状态机中，使得当前的控制层无法获得其它层的状态信息。每一层行为的设计者都是相互独立的，可能对其他层的信息一无所知，也不可能预测到这些层与当前层的关系。因此，每一次增加新的行为层，都需要对原有的层级做出相应的调整。**3)包容结构**提高了系统的响应速度，但它的行为设计还是比较简单的，缺乏必要的推理和协商能力。一般对于一些简单的任务能够获得令人满意的结果，但对于复杂的动态任务，缺乏必要的规划能力。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 分布式结构 (Distribute Architecture)



1997年，Rosenblatt在移动机器人导航中提出了DAMN(Distributed Architecture for Mobile Navigation)结构。它由一组分布式行为模块和集中命令仲裁器、模式管理器、控制器等组成。

图2.7显示了DAMN的组织结构，在该机器人结构中，底层有许多行为模块组成，每个行为模块都对应了机器人控制的某一方面，或是为了完成某一特定的任务，各行为可并行发生。当系统运行时，机器人的行为模块，例如轨迹跟踪、避障等，将自己的投票发送到命令仲裁模块进行投票仲裁。该模块把所有投票融合起来产生最终的结果命令，并将该命令发送至机器人的控制器。每个行为模块被分配了一个权值来反映它在相应环境中的执行优先级，命令仲裁器则根据这些权值来融合各行为模块发来的投票。同时，存在一个模式管理器与命令仲裁器相连，模式管理器可以根据不同的给定环境，不同的任务来改变权值，进而来改变命令仲裁器的决定。

DAMN组织结构在不同的任务、环境状态下，各行为模块会表现出不同的输入输出关系，即通过分布式投票、集中仲裁且动态改变权值的方式实现变构，从而使DAMN结构表现出自组织能力。该结构突破了传统结构中行为控制模块的固定框架，具有良好的可扩充和自适应、自组织性能。但由于信息多集中于命令仲裁器部分，随着行为模块的不断增多，容易造成信息的堵塞，所以该结构中还需要设计较为细致的命令融合处理函数。

1998年，Piaggio提出一种称为HEIR (Hybrid Experts in Intelligent Robots) 的非层次结构，如图2.8所示。这种非层次结构由处理不同类型知识的三个部分组成：符号组件 (S)、图解组件 (D) 和反应组件 (R)，每个组件都是一个有多种具有特定认识功能的、可以并发执行的智能体构成的专家组，各

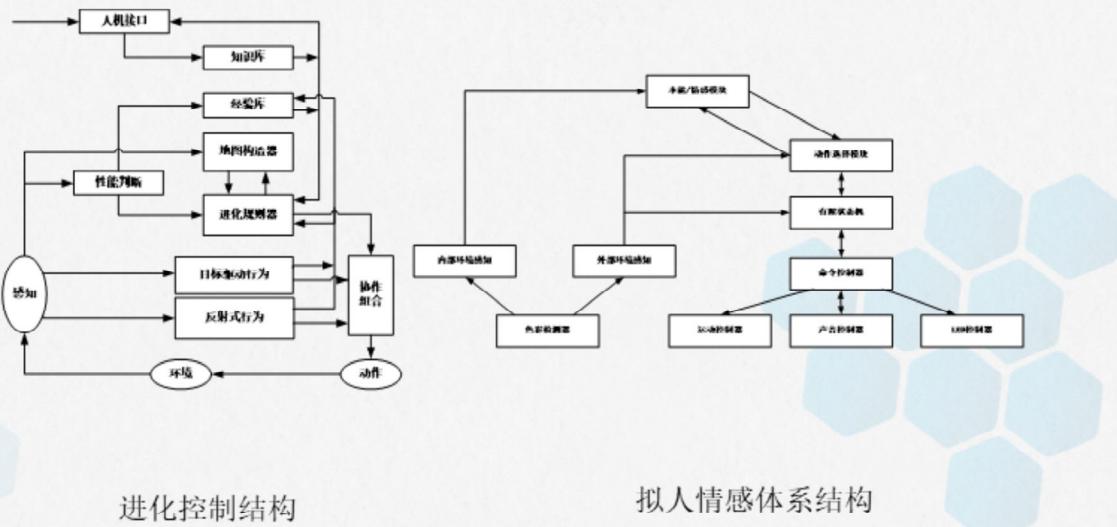
组件没有层次高低之分，自主地、并发地工作，相互通过信息交互进行协调。这其中，多智能体技术(Multi-agent)就是典型的分布式结构。多智能体技术的出现为设计复杂智能控制系统提供了新的思路，其中心思想是将大的复杂系统分解为小的、相对独立的子系统。依赖这些子系统彼此之间的竞争和协作来完成高智能化作业任务。

分布式结构突破了以往智能机器人软件体系结构中层次框架的分布模式，较好地解决了实时性问题。分布式结构的优势使它成为体系结构设计的一种良好选择。该结构中的智能体或模块具有极大的自主性和良好的交互性，可以独立求解局部问题并与系统中其它智能体或模块通过交互保持协调。但如何进行协调和统一的机制是一个重点，也是一个难点，处理不当可能出现冲突，使系统紊乱。分布式结构更适用于多机器人群体，例如目前流行的基于智能体（Agent）的多机器人技术。机器人单体采用分布式结构，则需要建立必要的集中协调机制，例如基于符号推理和行为相结合的协调机制，以及进化协调机制等。在实际设计过程中，可以采用一些结构的组合，例如分层分布式，将系统分层，在每一层中又采用分布式结构。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 仿生结构 (Biomimetic Architecture)



在仿生机器人的软件体系结构中，最具代表性的体系结构有进化控制结构和拟人情感模型等类型。典型的进化控制结构如图2.9所示，它将进化计算理论与反馈控制理论相结合，形成了一种新的智能控制方法—进化控制，它能很好地解决移动机器人的学习与适应能力等方面的问题。整个体系结构重点包括进化规划和基于行为的控制两大模块。进化规划可以根据知识库、经验库对周围环境以及自身的现状进行判断，及时地调整自身的目标和行为，以达到及时适应环境和在复杂的环境下完成任务的目的，具有良好的自主性、适应性。

而拟人情感体系结构是Arkin等在2001年提出的一种基于动物行为和情感作用的模型，并成功应用在机器狗AIBO的控制系统中，其体系架构见图2.10所示。其特色之处在于通过对人类情感的表现和他们与动物互动行为的研究，将情感与本能的行为加入到智能机器人的软件体系结构中，使机器人具有更好的交互功能。该模型主要由三个模块组成，包括释放（Releasing Mechanism）、激励（Motivation Creator）和行动选择模块（The Action Selection Module）。释放模块通过对环境的感知结果来计算释放，例如与认识目标之间的距离。激励模块是利用本能/情感模型来计算并维持机器狗的饥饿、口渴、疲倦、排泄、喜欢和好奇等本能和情感变量。最后根据前面两个模块综合选择机器狗的动作。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 体系结构中的任务规划

在人工智能中，规划（Planning）被定义为关于要采取行动的决策。而在机器人学中，规划包含任务规划(Task Planning)以及运动规划(Motion Planning)等内容。

### ➤ 基于状态空间的机器人任务规划

- 基本思想也是用“状态”和“操作”来表示和求解任务规划问题
- 状态空间(State Space)用来描述一个问题的全部状态以及这些状态之间的相互关系
- 基于状态空间的问题求解过程实际上是一个搜索过程
  - 如果用一个赋值的有向图来表示状态空间，可以采用基于广度优先搜索、基于深度优先搜索以及启发式搜索等图搜索算法进行问题求解，这些基本的图搜索算法将在第5章运动规划进行介绍。

从以上多种机器人软件体系结构中可以看出，机器人的智能主要体现在规划层的算法当中，无论是层次结构中的规划层或者慎思层，还是包容结构中的行为推理与规划层等，这些层主要体现机器人如何进行推理与决策，如何通过人-机-环境交互得到需要执行的行动方案的过程。

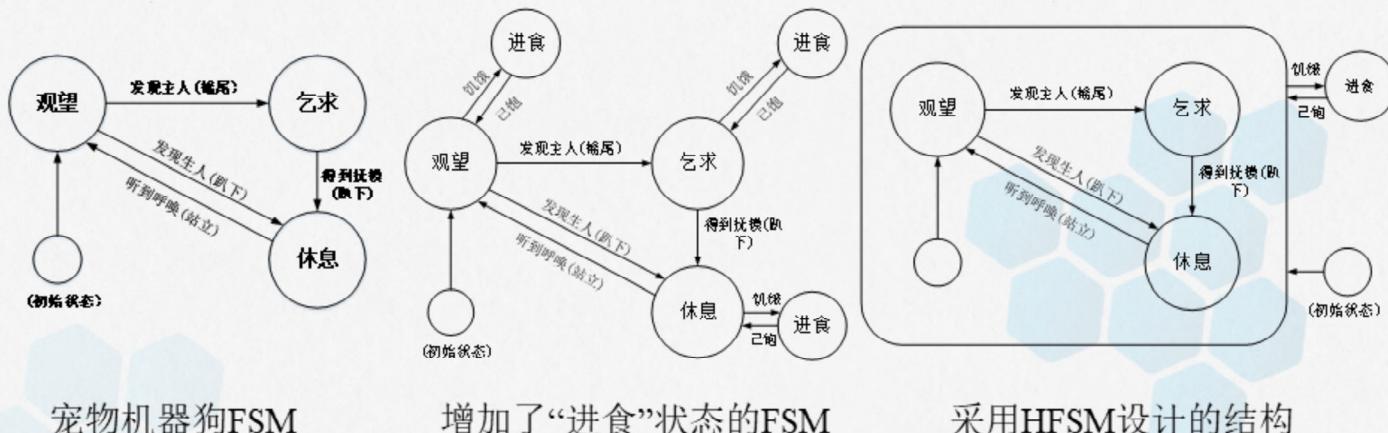
在人工智能中，规划（Planning）被定义为关于要采取行动的决策。而在机器人学中，规划包含任务规划(Task Planning)以及运动规划(Motion Planning)等内容。本节介绍机器人的任务规划（高层规划，High-Level Planning），而运动规划（低层规划，Low-Level Planning）则安排在第5章进行介绍。

# 智能机器人体系结构

Architecture of Intelligent Robots

## 体系结构中的任务规划

### 层次有限状态机(HFSM, Hierarchical Finite State Machine)



例2.1 设计一款宠物机器狗，实现如下的行为：

如果没有主人，就站立观望；

在站立观望时，如果发现主人，就摇尾表示乞求抚摸，如果发现是生人，就趴下休息；

如果在摇尾乞求时，得到抚摸，就趴下休息；

在趴下休息时，如果听到呼唤，就站立回到观望状态。

根据以上状态以及状态之间的转移条件描述，采用方向图设计的有限状态机如图2.11所示。在方向图中，圆代表状态，状态之间带箭头的线代表状态之间的转移，箭头的方向代表转移方向，线上标注的为事件和动作。

图2.11也可以用状态表进行表达，如表2.1所示。该表的第一行和第一列代表事件和状态，单元格内包含将要转移的次态和动作。（可以板书）

有限状态机的设计会随着状态的增加，导致状态之间的迁移变得复杂而混乱。为了解决有限状态机的这些缺点，研究人员提出了层次有限状态机(HFSM, Hierarchical Finite State Machine)。层次有限状态机中一个状态可以包含一个或者多个子状态。包含两个或多个状态的状态称为超态（Super States）。超态间的跳转隔离了一些不相关的状态，降低了整体结构的复杂性。每个超级状态都将一个子状态标识为启动状态，当跳转到该超态就从启动状态开始执行。相比于有限状态机，层次有限状态机做到了无关状态间的隔离，降低了状态机的复杂度。

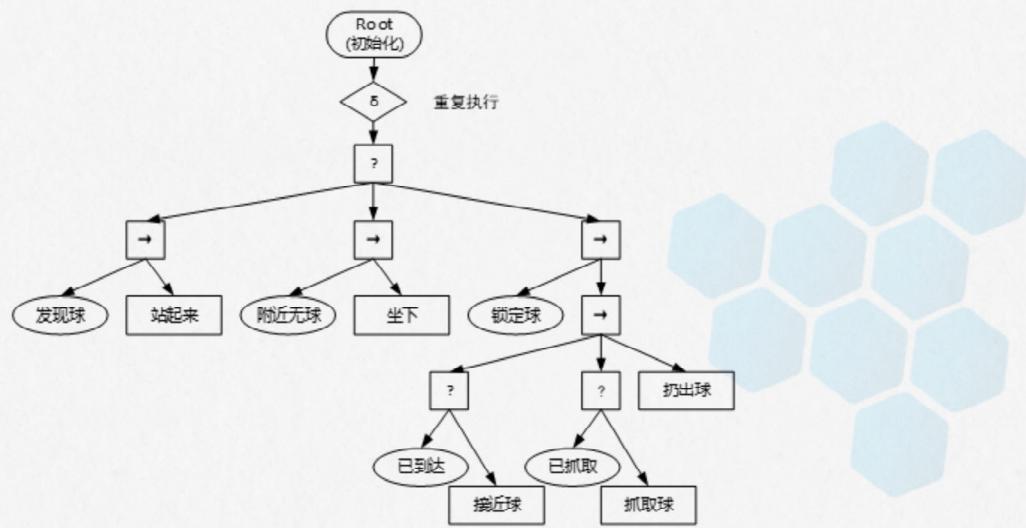
设想如果需要扩展一下上面宠物机器狗的行为，让宠物机器狗能在过一段时间感到饥饿的情况下舔一下骨头，从而进入到进食状态，如果采用FSM架构进行设计，就需要在每个状态都需要设计转移，如图2.12(a)所示，而采用图

2.12(b)所示的层次有限状态机设计宠物机器狗的架构就会更简洁有效一些。

## 智能机器人体系结构

Architecture of Intelligent Robots

## 行为树(Behavior Tree)

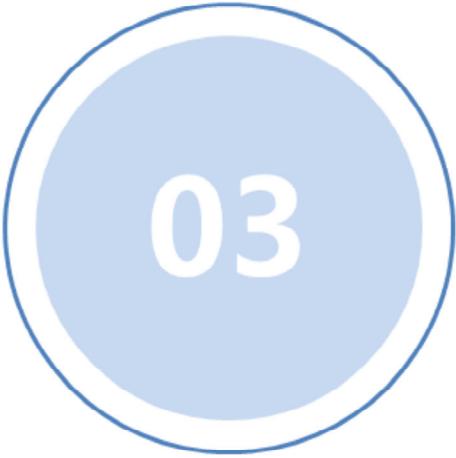


行为树（Behavior Trees, BTs）的概念最早由电子游戏领域提出，作为一个提高游戏角色控制结构模块化的工具被广泛使用。之后研究者发现BTs非常适用于构建机器人的控制架构：游戏中的角色与机器人非常相似，都是完成一些指定的任务并对外部变化及时响应。

(BT的基本介绍可以板书)

例2.2 采用行为树设计一款扔球机器人。当机器人附近没有小球时，就执行坐下动作；当机器人发现附近有小球时，则站起来准备去扔球；当锁定球后则跳入超级状态执行扔球的动作。依据行为树设计的机器人架构如图2.14所示，该机器人采用基于行为树的体系结构，行为树中不同节点的结合也可以实现各种状态的跳转。

以上介绍了设计智能机器人可以选择的多种软件体系结构类型，在设计机器人总体结构的时候，应根据实际系统的要求、特点和开销进行合理的选择。



03

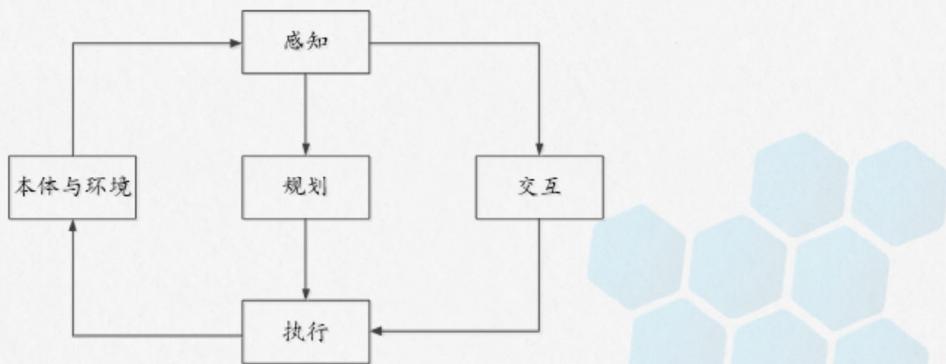
## 机器人系统的组成

分析机器人系统组成的目的，是为了能够从系统宏观的角度来对机器人系统进行抽象建模与分析。特别是机器人本体与环境的建模与分析方法，是进行感知、交互、规划与执行等系统设计的基础。

# 机器人系统的组成

System of Robots

## 机器人系统的体系结构组成



根据智能机器人体系结构的三种基本范式（慎思式、反应式和混合式），以及后续的研究与发展，我们可以将一个机器人系统的体系结构抽象成机器人本体与环境、感知、交互、规划、执行等部分组成。

机器人本体可能是工业机械臂、移动机器人、飞行机器人、人形机器人、仿生结构机器人等类型，或具有不同结构特征组合的复合机器人，例如移动机械臂等形式。

环境即指机器人所处的周围环境。环境不仅由几何条件（可达空间）所决定，而且由环境和它所包含的每个事物的全部自然特性所决定。机器人的固有特性，由这些自然特性及其环境间的互相作用所决定。在环境中，机器人会遇到一些障碍物和其他物体，它必须避免与这些障碍物发生碰撞，并对这些物体发生作用。环境信息一般是确定的和已知的，但在许多情况下，环境具有未知的和不确定的性质。

感知定义为机器人对自身与环境信息的获取，其感知系统包含内部传感器和外部传感器，内部传感器用于检测机器人自身的状态信息，如关节的位置、速度等变量。外部传感器用于检测机器人与环境之间的一些状态信息，包括距离、听觉、视觉、触觉等类型传感器。

交互指人、机、环境之间的互相作用，机器人可通过文字、语音、视觉、触觉等多种方式进行人、机、环境之间的交互。人、机、环境之间的交互随着传感技术以及人工智能技术的发展，正由单一类型的交互转向多传感器信息融合的多模态交互。

规划指机器人通过人-机-环境交互得到需要执行的行动方案的过程，它包含这一过程的问题求解技术。

执行定义为机器人在完成行动方案的过程中，和环境交互，实现任务目标的一系列动作，该动作可能是形体操作，例如移动、夹持、搬运、放置等，也可能是文字回复、语音回复等行为。

# 04

## 机器人位形与位形空间

机器人学的理论基础就是为了研究机器人在空间域、时间域的表征以及求解问题，在n维空间中描述机器人的位形及其与时间之间的关系，表达为机器人的运动学，而如果还需在时间域描述产生机器人位形变化所需力和力矩，则表达为机器人的动力学。

## 机器人坐标系与位姿

Coordinate system and Pose of Robot

分析刚体运动，我们首先需要确定参考坐标系。机器人参考坐标系，通常使用的有：

- **三维空间坐标系(笛卡尔坐标系)** (Cartesian coordinate system, 也称直角坐标系)。

除此之外，我们还会用到下面的坐标系：

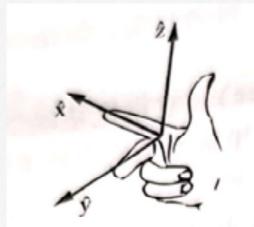
- **齐次坐标系** (Homogeneous Coordinates System)
- **指数坐标**(Exponential coordinate)
- **普吕克坐标系**(Plücker Coordinates)



机器人位姿(Pose)，即位置(position)和姿态(Orientation)

## 机器人坐标系与位姿

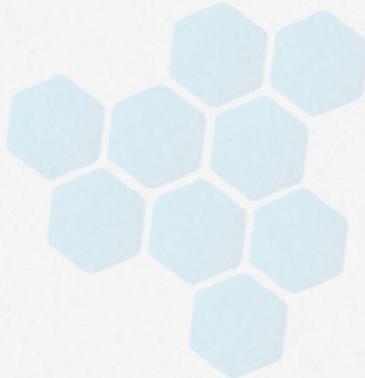
Coordinate system and Pose of Robot



(a) 右手坐标系法则



(b)右手坐标系的正向运动定义



如果将机器人抽象为空间中的刚体，将其质心看作是一个质点，那么，我们就可以将机器人的运动简化为质点在空间中的刚体运动。

分析刚体运动，我们首先需要确定参考坐标系。机器人相对于固定参考系的运动与时间之间的关系，通常使用三维空间坐标系(笛卡尔坐标系，Cartesian coordinate system，也称直角坐标系)以及空间物体的位姿来描述机器人的运动。对于平面移动机器人来说，我们通常只考虑平面坐标系下的移动，平面移动机器人的运动仅描述机器人位置与时间之间的关系；而对于空间机器人（包括飞行机器人、水下航行机器人、工业机械臂等）来说，则要考虑三维坐标环境下的机器人位姿(Pose)，即位置(position)和姿态(Orientation)。

三个坐标轴满足右手定则，则可得到三维的直角坐标系，见图2.17。其z-轴与x-轴、y-轴相互正交于原点。在三维空间的任何一点P的位置，可以用直角坐标 $\{x, y, z\}$ 来表达其位置。

## 机器人位形

Configuration of Robot

# 机器人坐标系与位姿

Coordinate system and Pose of Robot

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & a_1 \\ 0 & 1 & 0 & b_1 \\ 0 & 0 & 1 & c_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} \mu_x & v_x & w_x & p_x \\ \mu_y & v_y & w_y & p_y \\ \mu_z & v_z & w_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} R_{3 \times 3} & P_{3 \times 1} \\ f_{1 \times 3} & w_{1 \times 1} \end{bmatrix} = \begin{bmatrix} \text{旋转矩阵} & \text{位置矢量} \\ \text{透视矩阵} & \text{比例系数} \end{bmatrix}$$

$$R_{3 \times 3} = \begin{bmatrix} \mu_x & v_x & w_x \\ \mu_y & v_y & w_y \\ \mu_z & v_z & w_z \end{bmatrix}$$

## 机器人坐标系与位姿

Coordinate system and Pose of Robot

$$T = \begin{bmatrix} \mu_x & v_x & w_x & p_x \\ \mu_y & v_y & w_y & p_y \\ \mu_z & v_z & w_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T^{-1} = \begin{bmatrix} R_{3 \times 3}^{-1} & -(\bar{\mu})^T \bar{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} \bar{p} &= p_x \bar{i} + p_y \bar{j} + p_z \bar{k} \\ \bar{\mu} &= \mu_x \bar{i} + \mu_y \bar{j} + \mu_z \bar{k} \\ \bar{v} &= v_x \bar{i} + v_y \bar{j} + v_z \bar{k} \\ \bar{w} &= w_x \bar{i} + w_y \bar{j} + w_z \bar{k} \end{aligned}$$

姿态较常用的表示方法有：

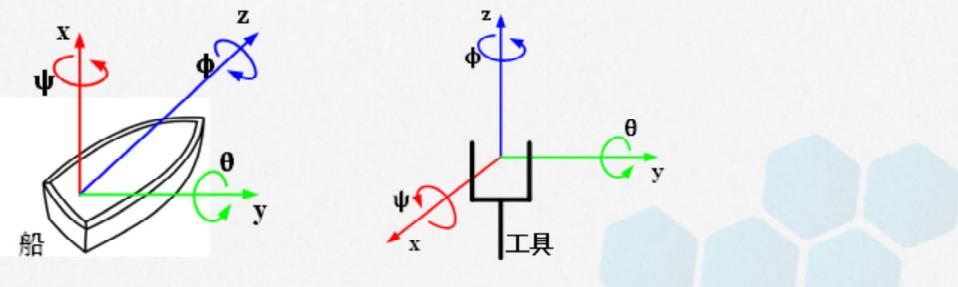
- **RPY角法**（滚动角、俯仰角、偏航角）
- 欧拉角法
- 罗德里格斯公式
- 四元素法
- 罗德里格斯参数法等方法

$T$ 矩阵反映了 $\Sigma O'$ 在 $\Sigma O$ 中的位置和姿态，即表示了该坐标系原点和各坐标轴单位矢量在固定坐标系中的位置和姿态。

如果需要求解 $\Sigma O$ 在 $\Sigma O'$ 中的位置和姿态，此时的齐次变换矩阵为，即求逆矩阵： $T^{-1}$

位置矩阵比较直观，代表动坐标系 $\Sigma O'$ 坐标原点在固定参考坐标系 $\Sigma O$ 中的位置，而姿态矩阵(altitude matrix)则有多种表示方法。姿态矩阵较常用的表示方法有**RPY角法**（滚动角、俯仰角、偏航角）、**欧拉角法**、**罗德里格斯公式**、**四元素法**以及**罗德里格斯参数法**等方法。

## RPY角法



$$RPY(\varphi, \theta, \psi) = Rot(z, \varphi)Rot(y, \theta)Rot(x, \psi)$$

$$\begin{aligned}
 R &= Rot(z, \varphi)Rot(y, \theta)Rot(x, \psi) \\
 &= \begin{bmatrix} C\varphi & -S\varphi & 0 \\ S\varphi & C\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\theta & 0 & S\theta \\ 0 & 1 & 0 \\ -S\theta & 0 & C\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\psi & -S\psi \\ 0 & S\psi & C\psi \end{bmatrix} \\
 &= \begin{bmatrix} C\varphi C\theta & -S\varphi S\psi + C\varphi S\theta S\psi & C\varphi S\theta C\psi + S\varphi S\psi \\ S\varphi C\theta & C\varphi S\theta S\psi + C\varphi S\psi & -C\varphi S\psi + S\varphi S\theta C\psi \\ -S\theta & C\theta S\psi & C\theta C\psi \end{bmatrix}
 \end{aligned}$$

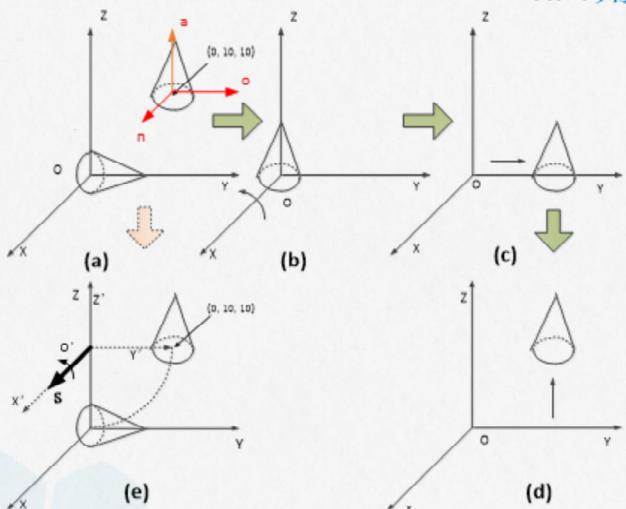
式(2-2)中, RPY表示横滚(roll)、俯仰(pitch)和偏转(yaw)三个旋转的组合变换。也就是说,先绕x轴旋转角 $\psi$ ,再绕y轴旋转角 $\theta$ ,最后绕z轴旋转角 $\phi$ ,其旋转矩阵的变换形式为依次左乘。采用RPY角法表示的姿态,其参考坐标系一般为全局坐标系。

## 机器人位形

Configuration of Robot

## 姿态表示方法

### RPY角法



刚体运动实例示意图

$$\begin{aligned}
 M &= \begin{bmatrix} n_x & o_x & a_x & 0 \\ n_y & o_y & a_y & 0 \\ n_z & o_z & a_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(x, n) & \cos(x, o) & \cos(x, a) & 0 \\ \cos(y, n) & \cos(y, o) & \cos(y, a) & 0 \\ \cos(z, n) & \cos(z, o) & \cos(z, a) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \\
 T &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_\theta & -s_\theta & 0 \\ 0 & s_\theta & c_\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} M \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot
 \end{aligned}$$

回顾一下在线性代数课程中所学的知识。在齐次坐标系下，可以将三维空间的刚体运动归纳为两种运动的组合，即绕某个主轴的平移运动或旋转运动。例2.3如图2.18所示，三维空间中的一个圆锥体，图(a)中表示需要将它从水平状态运动到垂直状态，这样的一个运动过程可以通(a)->(b)->(c)->(d)的一系列旋转、平移运动合成。更新后的圆锥体位姿可以通过初始位姿矩阵M依次左乘相应的旋转和平移矩阵即可得到。假设初始状态下，圆锥体的尖角指向为接近矢量a的方向，方向矢量o的方向与全局坐标系z轴方向相反，法向矢量n的方向与全局坐标系的x轴一致，初始位姿矩阵M以及经过齐次变换运算之后得到的更新后的位姿矩阵T的求解过程如下：

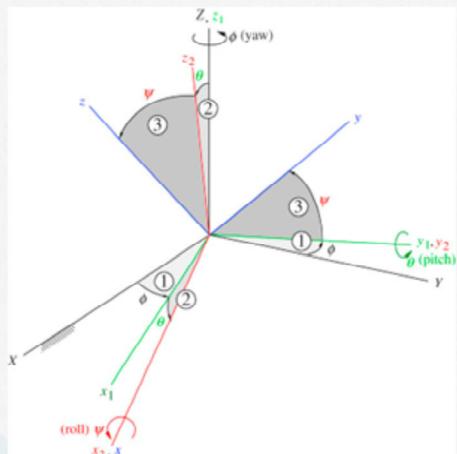
在上面的运算过程中，位姿的计算采用了齐次坐标系下的绝对变换，初始位姿矩阵依次左乘旋转、平移矩阵，得到更新后的位姿矩阵。如果只是求姿态，也可以采用RPY角法进行表达，初始状态下，其姿态角相当于绕x轴进行了-90度的偏转， $\psi = -90$ ,  $\theta = 0$ ,  $\phi = 0$ , RPY( $\phi, \theta, \psi$ ) =

$$Rot(z, \phi)Rot(y, \theta)Rot(x, \psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} =$$

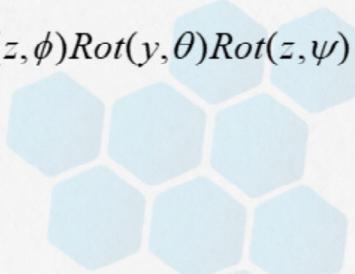
$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$ 。终止状态下，圆锥体在全局坐标系下较初始状态仅进行了绕x

轴90度的偏转， $\psi = 0$ ,  $\theta = 0$ ,  $\phi = 0$ , RPY姿态矩阵为 $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ 。

### 欧拉角法



$$Euler(\phi, \theta, \psi) = Rot(z, \phi)Rot(y, \theta)Rot(z, \psi)$$



$$R_{zyz} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

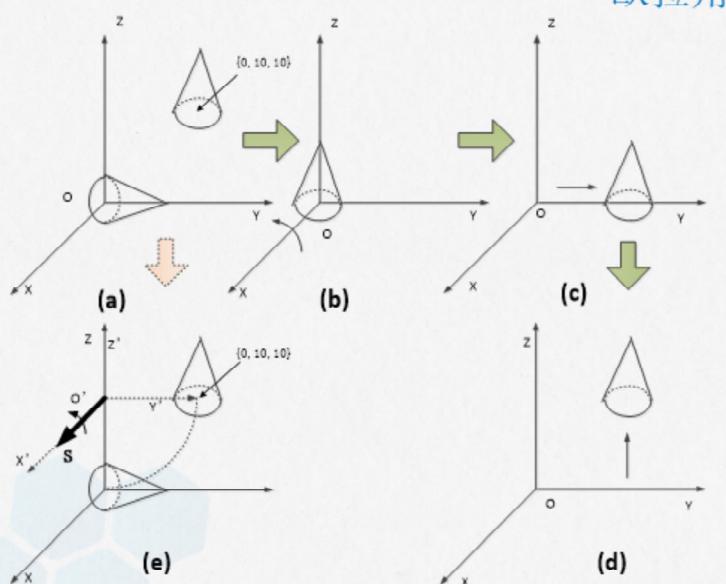
**RPY**角法通常是在全局固定坐标系下来表示刚体运动的姿态，而针对相对坐标系下的姿态，通常用欧拉角来进行描述。常用的欧拉角法控制顺序为：先绕z轴旋转 $\phi$ 角，再绕新的y轴 $y'$ 旋转 $\theta$ 角，最后绕新的z轴 $z''$ 旋转 $\psi$ 角来描述任何可能的姿态。

欧拉变换（Euler）也可由连乘三个旋转矩阵来求得，但其顺序为依次右乘。

**RPY**角法和欧拉角法这两种方法都是以一定顺序绕主坐标轴旋转三次来得到运动后的姿态描述，不同之处在于**RPY**角法是绕全局固定坐标系的主轴旋转，而欧拉角法则是绕动作标系的主轴旋转。实际上，采用绕主轴进行刚体的姿态变换表达方法中，按照三次旋转轴的顺序，三轴全用的有6种（ $xyz$ ,  $yzx$ ,  $zxy$ ,  $xzy$ ,  $zyx$ ,  $yxz$ ）表达，只用两轴的也有6种（ $zxz$ ,  $xyx$ ,  $yzy$ ,  $yzx$ ,  $xzx$ ,  $yxy$ ）表达，这些旋转如果始终绕固定轴，称为外旋，如果绕动轴，则称为内旋，因此总共可以有24种旋转顺序排列，但在机器人的姿态表示方法中，欧拉角法常采用 $zyz$ 或 $zyx$ 这两种表示方法。

## 机器人位形

Configuration of Robot



## 欧拉角法

## 姿态表示方法

终止状态下的姿态矩阵

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

终止状态到初始状态转换，可以看成：

- 绕z轴旋转 $\alpha = 0$ 度
- 再绕新的y轴 $y'$ 旋转 $\beta = 0$ 度
- 最后绕新的x轴 $x''$ 旋转了 $\gamma = -90$ 度

$$R_{zyx} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

在例2.3中，同样可以在圆锥体上附着物体坐标系，假设终止状态下物体坐

标系与全局坐标系方向一致，终止状态下的姿态矩阵为 $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ ，那么初

始状态下的姿态，可以看成绕z轴旋转 $\alpha = 0$ 度，再绕新的y轴 $y'$ 旋转 $\beta = 0$ 度，最后绕新的x轴 $x''$ 旋转了 $\gamma = -90$ 度。那么用欧拉角法表示的初始状态下的姿态矩阵为：

$$R_{zyx} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

## 机器人位形 Configuration of Robot

### 罗德里格斯公式

刚体在坐标系 $\{O'\}$ 中的初始位姿矩阵 $M'$

$$M' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -10 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

螺旋运动后的位姿矩阵 $T$ , 可以由指数积公式表达:

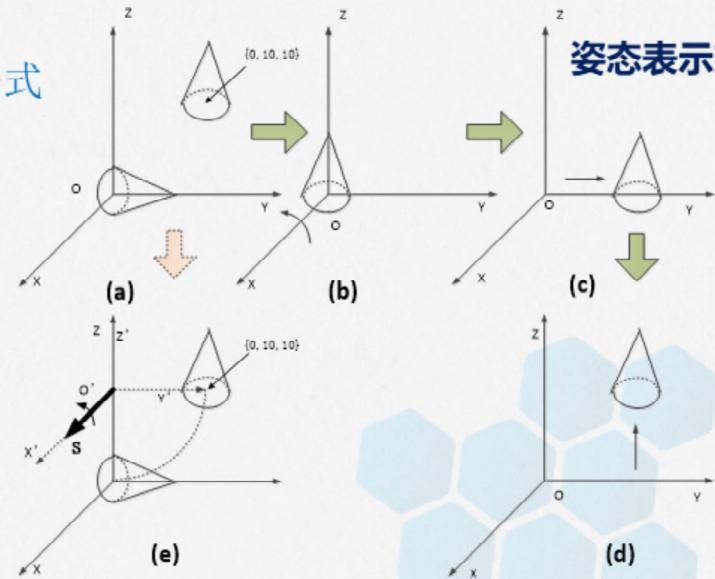
$$T' = e^{[s]\theta} M'$$

$$e^{[s]\theta} = \begin{bmatrix} \text{Rot}(\hat{\omega}, \theta) & v \\ 0 & 1 \end{bmatrix}$$

$$\text{Rot}(\hat{\omega}, \theta) = e^{[\hat{\omega}\theta]} = I + \sin \theta [\hat{\omega}] + (1 - \cos \theta)[\hat{\omega}]^2$$

这种固定坐标系及旋量轴的表示, 又称为**指数坐标**

Chasles-Mozzi定理: 任何刚体运动都可通过绕空间某一固定螺旋轴 $S$ 的运动来实现。



实际上, 在李群-李代数中, 这样的刚体运动还可以通过螺旋运动(screw motion)来进行描述, 即Chasles-Mozzi定理: 任何刚体运动都可通过绕空间某一固定螺旋轴 $S$ 的运动来实现。螺旋理论的基础要素可以追溯到Chasles和Poinsot在1800年代初期的工作, 使用Chasles和Poinsot定理作为起点, Robert S. Ball于1900年发表了完整的螺旋理论。

(板书计推导过程)

图2.18中的刚体运动, 它可以简单地通过绕某个坐标系 $\{O'\}$ 的螺旋轴 $S$ (screw axis)旋转一定角度 $\beta$ 而得到, 如图2.18(e)的直接旋转变换所示。该运动可以用四个螺旋参数坐标来表示, 即 $(\beta, O'_x, O'_y, O'_z)$ , 其中,  $(O'_x, O'_y, O'_z)$ 表示点 $O'$ 在 $\{O\}$ 坐标系中的坐标值。对于这个螺旋运动, 还可以用刚体绕 $O'$ 点的瞬时角速度 $\omega$ 和线速度 $v$ 来描述。

图2.18(e)中, 将坐标系 $\{O'\}$ 的 $x$ 轴定义和这个螺旋轴 $S$ 重合, 则刚体在坐标系 $\{O'\}$ 中的初始位姿矩阵为 $M'$ 。

$$M' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -10 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

该螺旋运动的正方向由右手法则确定, 该螺旋运动绕单一 $x'$ 轴的同方向, 所以 $(\omega_1, \omega_2, \omega_3) = (1, 0, 0)$ , 螺旋轴起始点建立在坐标系 $\{O'\}$ 的原点上, 所以 $(v_1, v_2, v_3) = (0, 0, 0)$ 。那么, 螺旋轴 $S = (\omega, v)$ 可写成:

$$\omega = (\omega_1, \omega_2, \omega_3) = (1, 0, 0)$$

$$v = (v_1, v_2, v_3) = (0, 0, 0)$$

螺旋运动后的位姿矩阵 $T$ , 可以由指数积公式表达:

$$T' = e^{[s]\theta} M' \quad (2-8)$$

经过Chasles-Mozzi定理证明, 给定任意的刚体 $P$ ,  $(R, p) \in SE(3)$ , 总能找到与之相对应的螺旋轴 $S = (\omega, v)$ 和标量 $\theta$ , 该矩阵对数满足:

$$e^{[s]\theta} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \quad (2-9)$$

式(2-9)用螺旋轴 $S = (\omega, v)$ 表示, 即为六参数向量的表达方式。

$$e^{[s]\theta} = \begin{bmatrix} Rot(\hat{\omega}, \theta) & v \\ 0 & 1 \end{bmatrix} \quad (2-10)$$

$$Rot(\hat{\omega}, \theta) = e^{[\hat{\omega}]\theta} = I + \sin \theta [\hat{\omega}] + (1 - \cos \theta)[\hat{\omega}]^2 \quad (2-11)$$

公式(2-11)又称为罗德里格斯(Rodrigues)公式。而这个 $Rot(\hat{\omega}, \theta)$ 本身如果只是绕某个固定轴的旋转, 它可以写成一个通用的表达式。

$$Rot(\hat{\omega}, \theta) = \begin{bmatrix} c_\theta + \hat{\omega}_1^2(1 - c_\theta) & \hat{\omega}_1\hat{\omega}_2(1 - c_\theta) - \hat{\omega}_3s_\theta & \hat{\omega}_1\hat{\omega}_3((1 - c_\theta) + \hat{\omega}_2s_\theta) \\ \hat{\omega}_1\hat{\omega}_2(1 - c_\theta) + \hat{\omega}_3s_\theta & c_\theta + \hat{\omega}_2^2(1 - c_\theta) & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_1s_\theta \\ \hat{\omega}_1\hat{\omega}_3((1 - c_\theta) - \hat{\omega}_2s_\theta) & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_1s_\theta & c_\theta + \hat{\omega}_3^2(1 - c_\theta) \end{bmatrix} \quad (2-12)$$

对于图2.18 (e)所示变换, 将式(2-10), (2-12)代入式(2-8)即可得到转换后的位姿:

$$T' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -10 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

该位姿为使用旋量指数积公式求解出来的坐标系 $\{O'\}$ 下的位姿, 将其转换到坐标系 $\{O\}$ 下, 即将坐标系进行一个平移变换即可:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} T' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

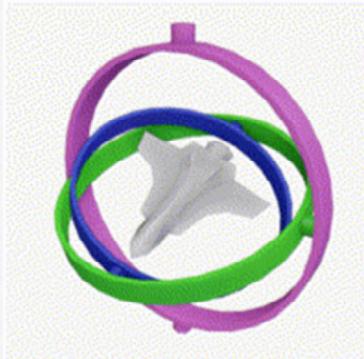
可以看出, 这个指数积形式的旋量法求解, 和前面齐次坐标系的组合变换是等效的。但这个指数积形式的旋量法更直观简明一些, 对于坐标系的建立, 只需建立一个初始坐标系, 然后找出各种运动对应的旋量轴 $S$ 即可。将这种固定坐标系及旋量轴的表示, 称之为指数坐标。式(2-8)为指数积公式, 可由此公式计算位姿矩阵。式(2-11)为罗德里格斯公式, 对应螺旋运动的旋转姿态矩阵。

## 机器人位形

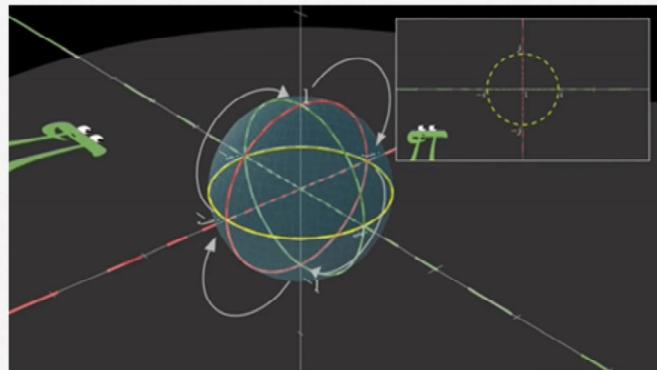
Configuration of Robot

## 姿态表示方法

### 万向节死锁 (Gimbal Lock)



### 四元数(Quaternion)



1843年，爱尔兰数学家William Rowan Hamilton创造了一个称为四元数(Quaternion)的超复数概念

$$q = (\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)n_x, \sin\left(\frac{\theta}{2}\right)n_y, \sin\left(\frac{\theta}{2}\right)n_z)$$

在前面我们描述的RPY或者欧拉角的姿态表示方法中，三个旋转顺序是固定的，我们可以通过这个旋转顺序得到所需的姿态。然而在某些特殊情况下，如果这其中的某两个旋转变换，由于旋转顺序及特定角度问题，导致可能变换的是同一外部轴，这样就会导致我们实际无法操纵某一个轴的自由度，从而导致所谓“万向节死锁 (Gimbal Lock) ”的产生。

1843年，爱尔兰数学家William Rowan Hamilton创造了一个称为四元数(Quaternion)的超复数概念。将三维空间坐标系中的一个向量  $\omega(n_x, n_y, n_z)$ ，在复平面内旋转  $\theta$  角，这个旋转向量我们使用四元数来进行表示：

$$q = (\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)n_x, \sin\left(\frac{\theta}{2}\right)n_y, \sin\left(\frac{\theta}{2}\right)n_z)$$

## 四元数(Quaternion)

$$q = (\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)n_x, \sin\left(\frac{\theta}{2}\right)n_y, \sin\left(\frac{\theta}{2}\right)n_z)$$

$$P = (0, x, y, z) = (0, S) \quad P \otimes Q = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = M(P)Q$$

$$P' = qPq^{-1}$$

$$P \otimes Q = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = M'(Q)P$$

将三维空间坐标系中的一个向量  $\omega(n_x, n_y, n_z)$ , 在复平面内旋转  $\theta$  角, 这个旋转向量我们使用四元数来进行表示:

$$q = (\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)n_x, \sin\left(\frac{\theta}{2}\right)n_y, \sin\left(\frac{\theta}{2}\right)n_z) \quad (2-10)$$

四元数表示的是复平面的旋转, 其旋转向量并无对应确切的物理含义。但我们可以将三维空间中的点  $P$  用虚四元数来表示:

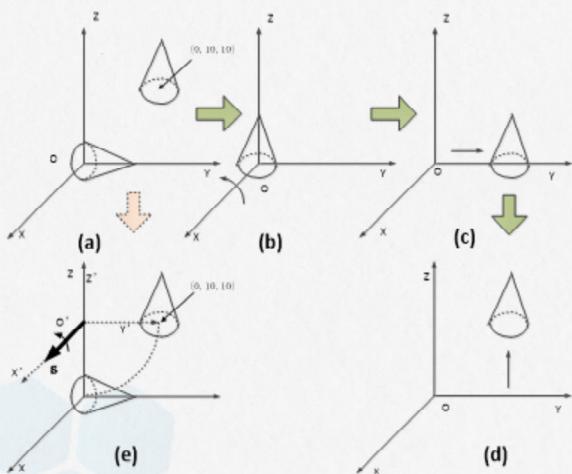
$$P = (0, x, y, z) = (0, S)$$

其中,  $S$  为旋转轴。

如果该点绕  $S$  轴旋转  $\theta$  角, 则该点新的旋转向量计算公式:

$$P' = qPq^{-1} \quad (2-11)$$

## 四元数(Quaternion)

 $S = (1, 0, 0)$  旋转90度

$P = (0, 0, 0, -10)$

$q = (\cos 45^\circ, \sin 45^\circ, 0, 0) = (\sqrt{2}/2, \sqrt{2}/2, 0, 0)$   
 $q^{-1} = \frac{q^*}{q * q} = \frac{q^*}{||q||^2} = q^* = (\sqrt{2}/2, -\sqrt{2}/2, 0, 0)$

$$\begin{aligned}P' &= qPq^{-1} \\&= \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0, 0\right)(0, 0, 0, -10)\left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, 0, 0\right) \\&= (0, 0, 10, 0) \\R_q &= \text{Quat}(\hat{\omega}, \theta) * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \\&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\end{aligned}$$

## 板书

在例2.3的图2.18(e)中，圆锥体底部中心点绕旋转轴 $S = (1, 0, 0)$ 旋转90度，旋转之后其坐标按式(2-13), (2-14)计算：

$P = (0, 0, 0, -10)$

$q = (\cos 45^\circ, \sin 45^\circ, 0, 0) = (\sqrt{2}/2, \sqrt{2}/2, 0, 0)$

模为1的四元数称为单位四元数，对于单位四元数， $||q|| = 1$ ，所以该四元数的逆 $q^{-1}$ 等于其共轭四元数 $q^* = (q_0, -q_1, -q_2, -q_3)$ ：

$q^{-1} = \frac{q^*}{q * q} = \frac{q^*}{||q||^2} = q^* = (\sqrt{2}/2, -\sqrt{2}/2, 0, 0)$

由式(2-15)得：

$P' = qPq^{-1} = \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0, 0\right)(0, 0, 0, -10)\left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, 0, 0\right) = (0, 0, 10, 0)$

可以得到更新后 $P'$ 点在坐标系 $\{O'\}$ 的坐标为 $(0, 10, 0)$ 。这个计算结果也是和前面的齐次坐标变换以及指数积形式的计算是等价的。

式(2-15)中的四元数乘法运算可以写成矩阵形式进行计算：

假设有两个四元数 $Q = (q_0, q_1, q_2, q_3)$ ,  $P = (p_0, p_1, p_2, p_3)$ , 则：

$$P \otimes Q = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = M(P)Q$$

或者：

$$P \otimes Q = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = M'(Q)P$$

需要注意的是四元数乘法不满足交换律。

采用四元数计算比直接使用旋转和平移的复合矩阵或者指数积公式的计算速度要更快，且这种方法对于计算内存的占用更低一些。

针对旋转轴S的刚体旋转运动，如果需要使用四元数来进行姿态矩阵的描述，可以用公式(2-16)进行表示。

$$\text{Quat}(\hat{\omega}, \theta) = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix} \quad (2-16)$$

由于 $\|Q\| = q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$ ，所以式(2-16)也可以写成：

$$\text{Quat}(\hat{\omega}, \theta) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (2-17)$$

针对例2.3，初始状态下的姿态矩阵为 $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$ ，则最终状态

下的姿态采用四元数进行验证：

$$R_q = \text{Quat}(\hat{\omega}, \theta) * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

由此可见采用四元数进行的姿态表达与前面描述方法结果是一致的。

## 四元数(Quaternion)

$$Quat(\hat{\omega}, \theta) = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}$$

$$Quat(\hat{\omega}, \theta) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

采用四元数计算比直接使用旋转和平移的复合矩阵或者指数积公式的计算速度要更快，且这种方法对于计算内存的占用更低一些。

针对旋转轴S的刚体旋转运动，如果需要使用四元数来进行姿态矩阵的描述，可以用公式（2-12）进行表示。

$$Quat(\hat{\omega}, \theta) = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix} \quad (2-12)$$

由于  $\|Q\| = q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$ ，所以式（2-12）也可以写成：

$$Quat(\hat{\omega}, \theta) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (2-13)$$

## 四元数(Quaternion)

## 罗德里格斯参数

四个参数:  $q_0, q_1, q_2, q_3$  $\rho$ : 三个参数

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1 \quad \rho = \hat{\omega} \tan \frac{\theta}{2} = (n_x \tan \frac{\theta}{2}, n_y \tan \frac{\theta}{2}, n_z \tan \frac{\theta}{2})$$

 $R(\rho_1, \rho_2, \rho_3)$ 

$$= \frac{1}{1 + \rho_1^2 + \rho_2^2 + \rho_3^2} \begin{bmatrix} 1 + \rho_1^2 - \rho_2^2 - \rho_3^2 & 2(\rho_1\rho_2 - \rho_3) & 2(\rho_1\rho_3 + \rho_2) \\ 2(\rho_1\rho_2 + \rho_3) & 1 - \rho_1^2 + \rho_2^2 - \rho_3^2 & 2(\rho_2\rho_3 - \rho_1) \\ 2(\rho_1\rho_3 - \rho_2) & 2(\rho_2\rho_3 + \rho_1) & 1 - \rho_1^2 - \rho_2^2 + \rho_3^2 \end{bmatrix}$$

$$\rho_i = \frac{q_i}{1 + q_0}, \quad (i = 1, 2, 3)$$

改进的罗德里格斯参数(Modified Rodrigues parameters):  $\rho = \hat{\omega} \tan \frac{\theta}{4} = (n_x \tan \frac{\theta}{4}, n_y \tan \frac{\theta}{4}, n_z \tan \frac{\theta}{4}) \quad 0 \leq \theta < 2\pi$ 

根据式 (2-17) 可得:

$$\rho = \hat{\omega} \tan \frac{\theta}{4} = (n_x \tan \frac{\theta}{4}, n_y \tan \frac{\theta}{4}, n_z \tan \frac{\theta}{4}) \quad (2-18)$$

得到的参数被称为改进的罗德里格斯参数(Modified Rodrigues parameters), 这个时候, 旋转角 $\theta$ 扩大到了 $0 \leq \theta < 2\pi$ 。

罗德里格斯参数只使用三个参数表示三维空间的旋转姿态, 在计算量上比四元数更有优势。

# 机器人位形

Configuration of Robot

## 姿态表示方法

姿态较常用的表示方法有：

➤ RPY角法（滚动角、俯仰角、偏航角）

$$RPY(\varphi, \theta, \psi) = Rot(z, \varphi)Rot(y, \theta)Rot(x, \psi)$$

➤ 欧拉角法

常用ZYX:  $Euler(\phi, \theta, \psi) = Rot(z, \phi)Rot(y, \theta)Rot(x, \psi)$

➤ 罗德里格斯公式

$$T' = e^{[s]\theta} M' \quad e^{[s]\theta} = \begin{bmatrix} Rot(\hat{\omega}, \theta) & v \\ 0 & 1 \end{bmatrix}$$
$$Rot(\hat{\omega}, \theta) = e^{[\hat{\omega}]\theta} = I + \sin \theta [\hat{\omega}] + (1 - \cos \theta)[\hat{\omega}]^2$$

➤ 四元素法

$$q = (\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)n_x, \sin\left(\frac{\theta}{2}\right)n_y, \sin\left(\frac{\theta}{2}\right)n_z)$$
$$Quat(\hat{\omega}, \theta) = \begin{bmatrix} 1 - 2(q_1^2 + q_2^2) & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & 1 - 2(q_1^2 + q_2^2) & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}$$
$$P = (0, x, y, z) = (0, S) \quad P' = qPq^{-1}$$

➤ 罗德里格斯参数法等方法

$$\rho = \hat{\omega} \tan \frac{\theta}{2} = (n_x \tan \frac{\theta}{2}, n_y \tan \frac{\theta}{2}, n_z \tan \frac{\theta}{2})$$
$$= \frac{R(\rho_1, \rho_2, \rho_3)}{1 + \rho_1^2 + \rho_2^2 + \rho_3^2} \begin{bmatrix} 1 + \rho_1^2 - \rho_2^2 - \rho_3^2 & 2(\rho_1\rho_2 - \rho_3) & 2(\rho_1\rho_3 + \rho_2) \\ 2(\rho_1\rho_2 + \rho_3) & 1 - \rho_1^2 + \rho_2^2 - \rho_3^2 & 2(\rho_2\rho_3 - \rho_1) \\ 2(\rho_1\rho_3 - \rho_2) & 2(\rho_2\rho_3 + \rho_1) & 1 - \rho_1^2 - \rho_2^2 + \rho_3^2 \end{bmatrix}$$
$$\rho_i = \frac{q_i}{1 + q_0}, \quad (i = 1, 2, 3)$$

# 机器人位形

Configuration of Robot

## 姿态表示方法

### 编程实例测试

相互之间的转换函数可以通过方向余弦矩阵采用齐次变换进行转换，课后请编写代码测试。

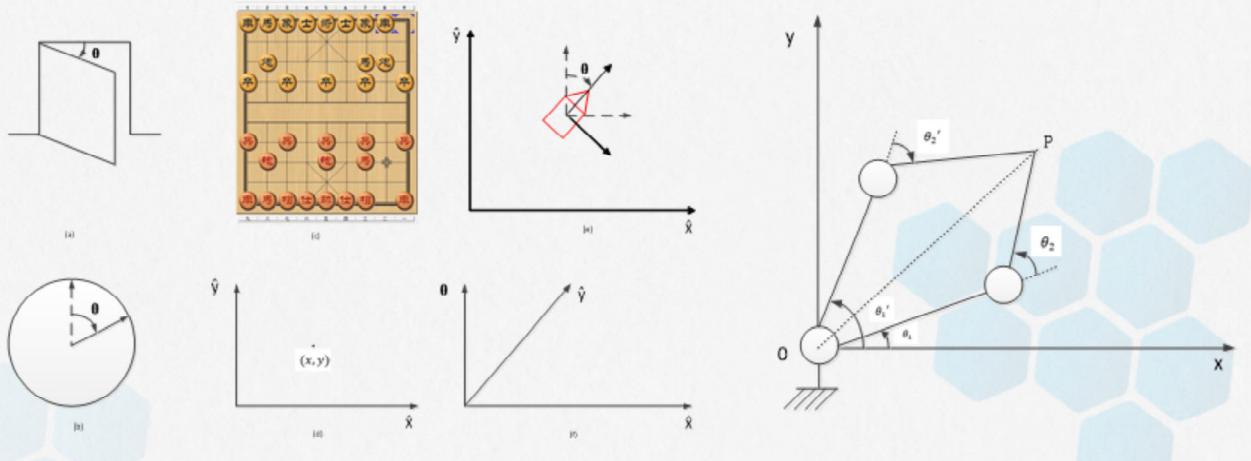
```
esir@esir-XPS-8920:~/robotics_exercise$ ./ch2
Initial POSE Matrix:      Screw POSE Matrix:
1 0 0 0                      1          0          0          0
0 1 0 0                      0  1.11022e-16   -1          10
0 0 1 0                      0          1  1.11022e-16 -1.11022e-15
0 0 0 1                      0          0          0          1
Eigen::Matrix<double,3,1> v, w,z Target POSE Matrix:      rotate vector by quaternion:
1 0 0 0                      1          0          0          0
0 0 -1 0                      0          0          0          0
0 1 0 0                      0          0          10         -1.77636e-15
0 0 0 1                      0          0          0          0
p[0] = w(0,0);                Target POSE Matrix:      rotate attitude by quaternion:
p[1] = w(1,0);                1          0          0          0
p[2] = w(2,0);                0  2.22045e-16   -1          0
what = w_hat(p);              0          1  2.22045e-16
z = Eigen::Matrix<double,3,1> z; Target POSE Matrix:      rotate vector by quaternion1:
void rotate_vector_by_quaternion(Eigen::Vector4d qstar;
Eigen::Vector4d qstar;
qstar << q(0), -q(1), -q(2), -q(3);
//Eigen::Vector4d p_prime;
//p_prime << 0, 0, -10;
Eigen::Matrix<double,4,4> MP;
MP << prime(0), prime(1), prime(0), -prime(0);
prime(1), prime(0), -prime(0), prime(0);
prime(0), prime(0), prime(1), prime(0);
prime(0), -prime(0), prime(0), prime(0));
Eigen::Vector4d tempV;
tempV = MP*qstar;
tempV = MP*tempV;
p = MP*tempV;
//std::cout<<"quaternion:\n"<<p<<std::endl;
}
1 0 0 0                      1          0          0          0
0 1 0 0                      0  1.11022e-16   -1          10
0 0 1 0                      0          1  1.11022e-16 -1.11022e-15
0 0 0 1                      0          0          0          1
esir@esir-XPS-8920:~/robotics_exercise$
```

参考代码: <https://github.com/mhuasong/Basics-of-Robotics-Theory-and-Technology/tree/main/ch2>

# 机器人位形

Configure of Robot

## 位形及位形空间 Configure of Robot and C-Space



机器人本体在所处环境中的位置，在现代机器人学中我们把它称之为位形（Configuration），包含所有可能的机器人位形的n维空间称为位形空间(C-Space)。

图2.15中，图2.15(a)门的位形可以通过单一参数：门的开合角度来表示，其位形空间为一维空间{1D ( $\theta$ )}。图2.15(c)中，中国象棋棋盘上的每个棋子，其位置可以用纵横格数来表示，其位形空间为二维空间{2D ( $x, y$ )}。图2.15(e)中，平面可移动小车，其位形除了纵横坐标外，还有小车的头部方向角，其位形空间为三维空间{3D ( $x, y, \theta$ )}。

在以上的表示方法中，我们将机器人抽象成形状已知的刚体，刚体的自由度表示了其位形所需的最小坐标数（或维度数）。

我们将机器人在执行任务时能够自然表达的空间称为任务空间（task space），而工作空间(work space)则是指机器人末端执行器所能到达的空间。

任务空间的所有点对于机器人来说，并不一定都可到达。而工作空间虽然表示机器人都可达的点，但对于工作空间中的点来说，其对应的机器人位形可能有多个。例如图2.16中所示，对于一个平面两连杆机构来说，对应P点的末端可达位形可以有两个位形表示{( $\theta_1, \theta_2$ ), ( $\theta'_1, \theta'_2$ )}。

# 机器人位形

Configure of Robot

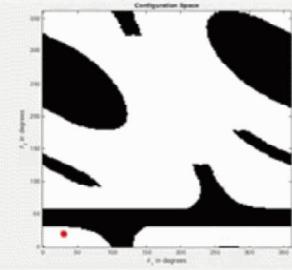
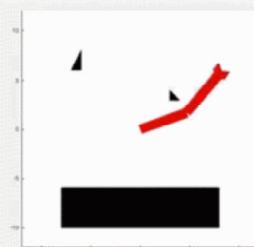
## 位形及位形空间

### Configure of Robot and C-Space

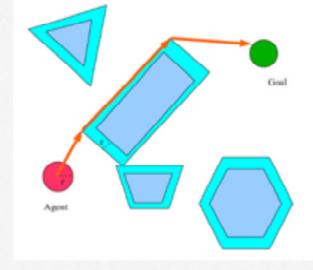
正解

逆解

$$\begin{Bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{Bmatrix}$$



$$\begin{Bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{Bmatrix}$$



$$C_{free} = C_{space} - C_{obstacle}$$

机器人的理论基础就是为了研究机器人在空间域、时间域的表征以及求解问题，在n维空间中描述机器人的位形及其与时间之间的关系，表达为机器人的运动学，而如果还需在时间域描述产生机器人位形变化所需的力和力矩，则表达为机器人的动力学。当已知机器人位形空间的初始位形，已知所有位形变量的变化步骤，求机器人最终所处的位形，这被称为机器人的正运动学，反之，已知机器人所要达到的位形，求机器人可以从初始位形到最终位形的变化步骤，这被称为机器人的逆运动学。同样，在时间域的求解问题，也存在机器人的正逆解问题。

除了机器人的运动学与动力学之外，我们还需要研究机器人的运动规划问题。例如，移动机器人在二维平面运动，机器人所处的作业空间我们把它称为任务空间，除去障碍物以外机器人不可达的空间（C-Obstacle），其余的空间我们称其为自由空间（C-Free）。如果不考虑该机器人的几何半径，把机器人抽象为空间中的一点， $C_{task}$ 就等于 $C_{space}$ ，则：

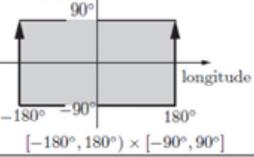
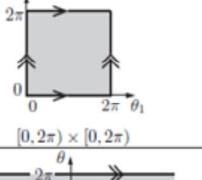
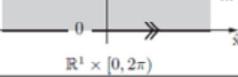
$$C_{free} = C_{space} - C_{obstacle} \quad (2-1)$$

式(2-1)中， $C_{free}$ 等同于机器人工作空间 $C_{work}$ ，机器人的轨迹规划问题即可表达为我们需要在 $C_{free}$ 中找到一个合适的解，例如最短路径、时间最优路径、能量最优路径等问题。

一般来说移动机器人的运动规划是在 $C_{free}$ 中求解，本身就考虑了避障问题，而工业操作臂(manipulator)安装于宽敞的作业环境，工作空间使用护栏隔离，其传统运动规划算法没有考虑避障问题，如果需要在狭窄的空间进行操作，则也应考虑避障问题。

# 机器人位形

Configure of Robot

system	topology	sample representation
point on a plane	$\mathbb{E}^2$	
spherical pendulum	$S^2$	
2R robot arm	$T^2 = S^1 \times S^1$	
rotating sliding knob	$\mathbb{E}^1 \times S^1$	

- Holonomic constraint
- Pfaffian 约束
- 可积约束(Integrable constraint)
- Nonholonomic constraint

C-Space: 拓扑 (Topology) 与拓扑等效(Topology equivalent)  
C-Space的维度问题以及形状问题同样重要。

一维空间: 圆、直线、线段, 圆表示为 $S^1$ 或者 $S^n$ , 直线表示 $\mathbb{E}$ ,

由于 $\mathbb{E}$ 上的一点通常用实数来表示, 因此通常表示成 $\mathbb{R}^1$ , 线段由于存在两个端点,  $[a,b] \subset \mathbb{R}^1$

$\mathbb{R}^n$ 表示n维欧氏空间,  $S^n$ 表示为n+1维空间内的n维球表面。

一些C-Space可以表示为两个或更多个低维空间的笛卡尔积 (Cartesian product)

参见MR教材P16-20页

05

## 机器人关节类型与自由度

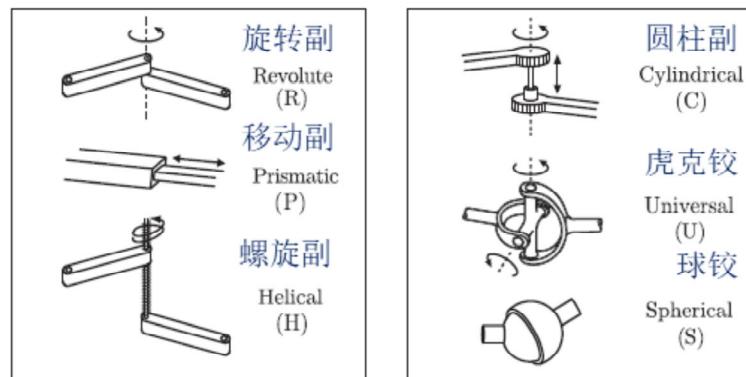
了解机器人常用关节类型，针对理论计算对机器人自由度进行分析。



## 机器人关节类型与自由度

Type of Robot Joint and Freedom

### Types of Joints



## ➤ Robots are serial “chain” mechanisms made up of

- “links” (generally considered to be rigid), and
- “joints” (where relative motion takes place)

## ➤ Joints connect two links

- Link 0 - **Joint 1** - Link 1 - **Joint 2** - Link 2 -

# 机器人关节类型与自由度

Type of Robot Joint and Freedom

## Types of Joints

表 2.2 常用关节的自由度  $f$  与约束数  $c$

关节类型	自由度 $f$	平面机构两刚体之间的 约束度 $c_p$	空间机构两刚体之间的 约束度 $c_s$
转动副	1 <sup>o</sup>	2 <sup>o</sup>	5 <sup>o</sup>
移动副	1 <sup>o</sup>	2 <sup>o</sup>	5 <sup>o</sup>
螺旋副	1 <sup>o</sup>	N/A(非平面机构) <sup>o</sup>	5 <sup>o</sup>
圆柱副	2 <sup>o</sup>	N/A(非平面机构) <sup>o</sup>	4 <sup>o</sup>
万向节	2 <sup>o</sup>	N/A(非平面机构) <sup>o</sup>	4 <sup>o</sup>
球铰	3 <sup>o</sup>	N/A(非平面机构) <sup>o</sup>	3 <sup>o</sup>



# 机器人关节类型与自由度

Type of Robot Joint and Freedom

## Grübler's formula for the DOFs of robot

Joint type	dof $f$	Constraints $c$ between two planar rigid bodies	Constraints $c$ between two spatial rigid bodies
Revolute (R)	1	2	5
Prismatic (P)	1	2	5
Helical (H)	1	N/A	5
Cylindrical (C)	2	N/A	4
Universal (U)	2	N/A	4
Spherical (S)	3	N/A	3

Grübler's formula for the number of degrees of freedom of the robot is

$$\begin{aligned} \text{dof} &= \underbrace{m(N-1)}_{\text{rigid body freedoms}} - \underbrace{\sum_{i=1}^J c_i}_{\text{joint constraints}} \\ &= m(N-1) - \sum_{i=1}^J (m-f_i) \\ &= m(N-1-J) + \sum_{i=1}^J f_i. \end{aligned}$$

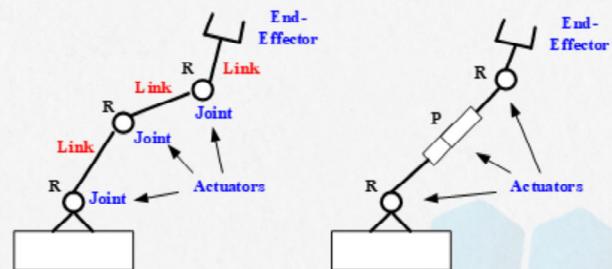
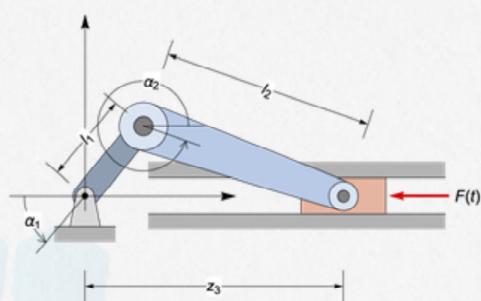
Grübler公式只有在所有关节约束都是独立的情况下才能成立。否则，该公式只能用于判断自由度的下限值。有关位形空间的约束与奇异相关内容将在运动学相关章节进行分析。

对于一个具有N个构件（含基座）的机构，令J为关节数，m为刚体的自由度数（对于平面机构，m=3；对于空间机构，m=6），fi为关节i对应的自由度数，ci为其对应的约束数，对于所有的i，始终满足fi+ci=m。则用于计算机器人自由度的Grübler公式可以写成：

上述公式只有在所有关节约束都是独立的情况下才能成立。否则，该公式只能用于判断自由度的下限值。有关位形空间的奇异相关内容将在运动学相关章节进行分析。

## 机器人关节类型与自由度

Type of Robot Joint and Freedom



$$dof = 3$$

如果是空间机构,  $m = 6$ ,  $dof = 6(4 - 1 - 3) + 3 = 3$

$n$ 轴串联机械臂的自由度计算结果为 $n$ :

$$dof = 6((n + 1) - 1 - n) + n = n$$

## 机器人关节类型与自由度

Type of Robot Joint and Freedom

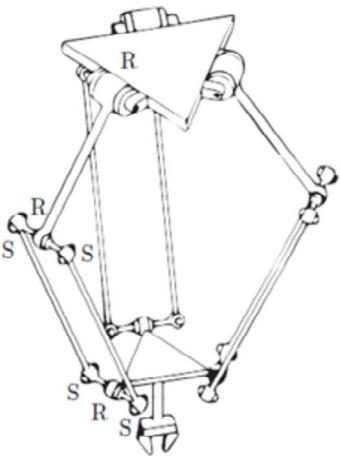


图2.26 Delta并联机器人

*Grübler*

$$dof = 6(17 - 1 - 21) + 9(1) + 12(3) = 15$$

实际上，这3个空间平行四边形闭链机构是为了确保动静平台始终保持平行设置的。计算出的15个自由度中，实际上末端执行器的运动只有3个直角坐标系下的平移自由度，其他的12个实际为局部自由度，这12个自由度反映了空间平行四边形结构中的12根杆（3个平行四边形）绕其长轴的扭转。

*Grübler*修正公式

$$dof = m(N - 1 - J) + \sum_{i=1}^J f_i + v - \xi$$

例2.6 如图2.26所示的Delta机器人，上面的静平台固定在基座上，通过连接在静平台上的三个转动副驱动3条支腿，带动下面的动平台运动。每条支腿都含有一个空间平行四边形闭链，其共有3个转动副、4个球铰、5根杆件，外加2个平台，因此，该机构一共有N=17个构件（3x5+2=17）、21个运动副（9个转动副、12个球铰），根据*Grübler*公式，计算其自由度：

$$dof = 6(17 - 1 - 21) + 9(1) + 12(3) = 15$$

实际上，这3个空间平行四边形闭链机构是为了确保动静平台始终保持平行设置的，计算出的15个自由度中，实际上末端执行器的运动只有3个直角坐标系下的平移自由度，其他的12个实际为局部自由度，这12个自由度反映了空间平行四边形结构中的12根杆（3个平行四边形）绕其长轴的扭转。

在空间机构学的研究中，对于复杂的机构，比如多环并联机构（具有两个或更多个闭链的并联机构），在其自由度的计算公式中需要考虑冗余约束以及局部自由度，在这种情况下，机构的自由度计算*Grübler*公式需要进行修正。

$$dof = m(N - 1 - J) + \sum_{i=1}^J f_i + v - \xi \quad (2-24)$$

式(2-24)中， $v$ 表示多环并联机构在去除公共约束因素后的冗余约束数， $\xi$ 为机构中存在的局部自由度。本章描述的*Grübler*公式(2-23)，只有在所有关节约束都是独立的情况下才能成立，对于复杂机构，只能提供机构的自由度下限值。

本节只介绍了适用于简单机构的机器人自由度计算与分析方法，实际上有关空间机构的自由度分析还有多种方法，例如：将机构的运动学方程构成一个矩阵，通过分析其秩来计算其自由度的方法；将机构的每一个闭链进行拆分，然后通过虚位移矩阵法来分析机构自由度的方法；通过Jacobian矩阵计算其

零空间，分析机构自由度的方法；基于群论、李代数、微分几何的知识来解决自由度计算的方法；以及基于螺旋理论进行自由度计算等方法。这部分相关的知识超出了本章的描述范围，有兴趣的读者请查阅空间机构学的相关文献。

针对理论计算对机器人自由度进行分析，其位形空间可能还存在奇异，有关奇异性以及工作空间的位形分析将在第3章运动学中进行描述。

# 06

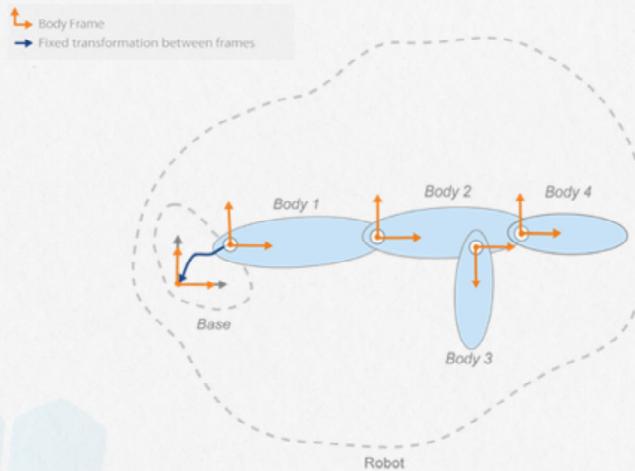
## 机器人建模描述文件与仿真

机器人的形式多种多样，但如果按照刚体的运动进行抽象，我们可以把机器人简化成刚体树(Rigid Body Tree)，刚体树由通过关节 (Joint) 连接的刚体 (Body) 组成。采用刚体树结构进行简化描述的机器人模型，我们称为刚体树机器人模型(Rigid Body Tree Robot Model)，目前有几种常用的机器人通用描述语言：URDF、SDF、MJCF等文件格式。

# 机器人建模描述文件与仿真

Robot Description and Simulation

## 刚体树机器人模型(Rigid Body Tree Robot Model)



### 关节(joint):

- 固定关节(fixed joint, 表示对应的刚体固定连接到父节点上, 无法运动)
- 旋转关节(revolute joint, 表示对应的刚体相对于父节点进行旋转运动)
- 平移关节(prismatic joint, 表示对应的刚体相对于父节点只能进行线性平移运动)

刚体树由基座(base)、刚体(rigid)、关节(joint)等组件组成

在早期的机器人学中，机器人大都指工业机器人：操作臂(manipulator)，那个时候的机器人被简化为连杆(link)和关节(joint)。操作臂几乎都是由刚性连杆组成，相邻连杆间可做相对运动的连接称为关节。关节按其运动形式分为：旋转关节和平移关节。

而现在机器人的形式多种多样，但如果按照刚体的运动进行抽象，我们可以把机器人简化成刚体树(Rigid Body Tree)，刚体树由通过关节(Joint)连接的刚体(Body)组成。采用刚体树结构进行简化描述的机器人模型，我们称为刚体树机器人模型(Rigid Body Tree Robot Model)。每个刚体都有一个关节，该关节定义了该刚体如何相对于其父节点在树中的运动(旋转或平移)。这样进行简化之后，我们就可以通过在每个关节上设置一个固定的变换(旋转或平移)，描述从一个关节到下一个关节的转换关系，进行运动学和动力学的分析。

刚体树由基座(base)、刚体(rigid)、关节(joint)等组件组成。

一般来说，每个刚体树都有一个基座(base)，我们一般将世界坐标系定义在基座上，它是刚体的第一个连接点。

刚体(rigid)由刚性主体创建，一般表示为无法变形的实体，单个刚体上任意两点之间的距离保持不变，刚体树由刚体链接而成。我们可以在刚体上定义该刚体的坐标系，并将它和所连接的关节进行关联。

每个刚体都有一个关节(joint)，该关节定义了该刚体相对于其父节点的运动。关节是连接机器人模型中两个刚体的连接点。如果机器人的单个物理部件上有多个关节或不同运动轴时，需要将这个物理部件采用多个刚体进行表达。

关节的类型包括固定关节(fixed joint, 表示对应的刚体固定连接到父节点上，

无法运动)、旋转关节(**revolute joint**, 表示对应的刚体相对于父节点进行旋转运动)、平移关节 (**prismatic joint**, 表示对应的刚体相对于父节点只能进行线性平移运动)。

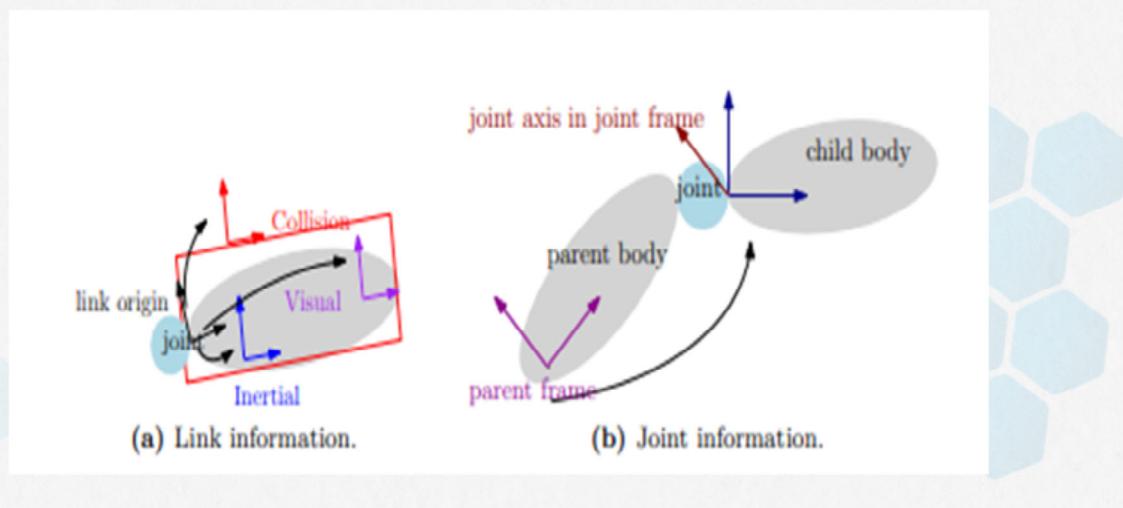
在图2.21中，刚体1固定在基座上，刚体1的坐标系和基坐标系之间是固定的变换关系。刚体1和刚体2之间是一种串联关系，刚体3和刚体4的父节点都是刚体2，它们之间是一种并联关系。

机器人所有的部件，包括末端执行器(**End effector**)，都可以抽象成刚体及刚体对应上一个父节点的运动部件（关节），其运动可等价为其上的坐标系对应其它参考坐标系的变换。这个时候我们可以根据需求，采用上一节中介绍的多种姿态变换矩阵进行求解。

# 机器人建模描述文件与仿真

Robot Description and Simulation

## URDF模型



URDF (Unified Robot Description Format, 统一机器人描述格式) 是ROS中使用的机器人模型描述格式，它使用XML (eXtensible Markup Language, 扩展标记语言) 格式来描述机器人模型，目前有很多机器人仿真控制软件都直接支持URDF文件解析。

URDF文件中常用的XML标签包含`<robot>`、`<link>`、`<joint>`等标签。`<robot>`是完整机器人模型的最顶层标签，一个完整的机器人模型由一系列连杆标签`<link>`和关节标签`<joint>`组成，`<link>`和`<joint>`标签都必须包含在`<robot>`标签内。

连杆`<link>`标签用于描述机器人某个刚体部分的物理及外观属性，包括惯性矩阵 (inertial matrix) 、外观参数(Visual properties)、碰撞参数 (Collision properties) 等。

`<inertial>`标签用于描述连杆(link)的惯性参数，`<visual>`标签用于描述机器人连杆(link)的外观参数，，而`<collision>`标签用于描述连杆(link)的碰撞属性。

# 机器人建模描述文件与仿真

Robot Description and Simulation

## URDF模型基本格式

```
<link name="my_link">
  <inertial>
    <origin xyz="0 0 0.5" rpy="0 0 0"/>
    <mass value="1"/>
    <inertia ixx="100" ixy="0" ixz="0" iyy="100" iyx="0" iyz="100" izz="100" />
  </inertial>

  <visual>
    <origin xyz="0 0 0" rpy="0 0 0" />
    <geometry>
      <box size="1 1 1" />
    </geometry>
    <material name="Cyan">
      <color rgba="0 1.0 1.0 1.0"/>
    </material>
  </visual>

  <collision>
    <origin xyz="0 0 0" rpy="0 0 0"/>
    <geometry>
      <cylinder radius="1" length="0.5"/>
    </geometry>
  </collision>
</link>
```

其中，`name`属性是必须的，表示连杆的名称。

`<inertial>`标签包括下面的所有属性都是可选的。`<origin>` 表示惯性参考坐标系相对于连杆参考坐标系的位姿。惯性参考坐标系的原点必须位于重心，惯性参考坐标系的轴无需与惯性主轴对齐。`Xyz`代表惯性坐标系相对于连杆坐标系的原点位置偏移值，缺省为向量0；`rpy`代表惯性坐标系相对于连杆坐标系RPY姿态角的弧度值，缺省为重合；`mass`表示连杆的质量。

接下来的`inertia`是一个3x3旋转惯性矩阵。

对于三维空间中任意一参考点K与以此参考点为原点的直角坐标系Kxyz，一个刚体的惯性张量I可以表示为下面的3x3的矩阵。

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

由于旋转惯量矩阵是对称的，因此在此处仅使用属性`ixx`, `ixy`, `ixz`, `iyy`, `iyz`, `izz`指定该矩阵的6个对角线上的元素。其中，`Ixx`, `Iyy`, `Izz`分别是绕x, y, z轴的转动惯量，`Ixy`, `Iyz`, `Ixz`为惯性积，即惯性坐标系中面积微元dA与其到相应指定的轴距离乘积的积分。

`<visual>`标签整体可选，其下可以标注几何和材料方面的属性，其中如果存在`<visual>`标签，其几何属性`<geometry>`必须标注，如果是长方体`<box>`, `size`属性为三个边长，坐标系原点位于长方体`box`的几何中心。如果是圆柱体`<cylinder>`，则需指明半径`radius`和长度`length`的值，坐标系原点位于圆柱体的几何中心。如果是球体`<sphere>`，则需指明半径`radius`，坐标系原点位于球体的几何中心。`<material>`属性整体可选，主要有颜色`<color>`等属性可以

指定。

**<collision>**标签整体可选，用于描述连杆的碰撞属性，在图2.22中，连杆link的**<collision>**区域应大于外观**<visual>**的区域，如果有物体与**<collision>**区域相交，就认为与该连杆发生了碰撞。

# 机器人建模描述文件与仿真

Robot Description and Simulation

## <visual>以及<collision>标签

```
<link name="base_link">
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0"/>
    <geometry>
      <mesh
        filename="package://robot_description/meshes/base_link.DAE"
      </geometry>
    </visual>
    <collision>
      <origin rpy="0 0 0" xyz="-0.065 0 0.0"/>
      <geometry>
        <mesh
          filename="package://robot_description/meshes/base_link_simple.DAE"
        </geometry>
      </collision>
      <inertial>
        ...
      </inertial>
    </link>
```



在<visual>以及<collision>标签中，如果我们需要加载更细致的外观与碰撞形体特征，则可以在几何属性<geometry>中标注成<mesh>，在该属性下直接指定用于标注几何属性的本地CAD文件。URDF的mesh文件通常有两类：包含材质颜色等信息的dae(Digital Asset Exchange)文件（非必须，DAE文件是面向交互式[3D](#)应用程序Collada的3D模型文件，可用于多个图形程序之间交换数字数据）与用于碰撞的stl文件(一种为快速原型制造技术服务的三维图形文件格式)，具体支持的CAD文件格式要求请参见相关仿真平台软件的说明。例如如下的实例格式：

# 机器人建模描述文件与仿真

Robot Description and Simulation

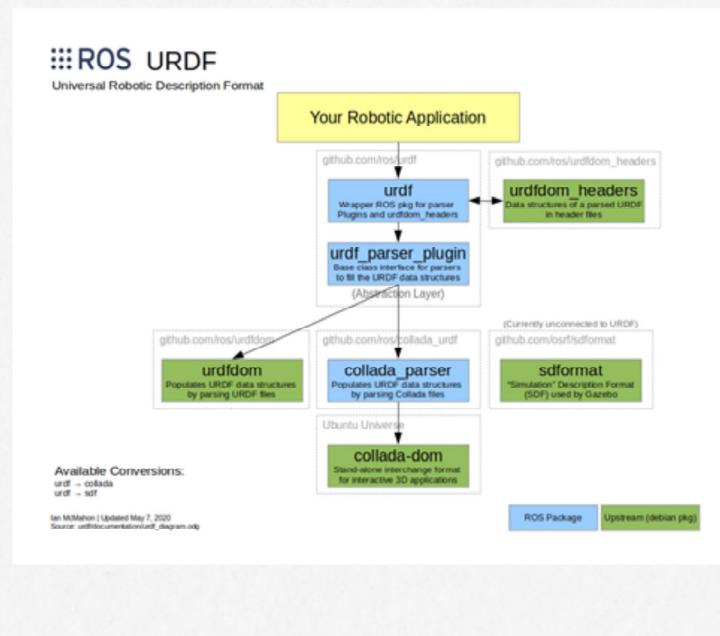
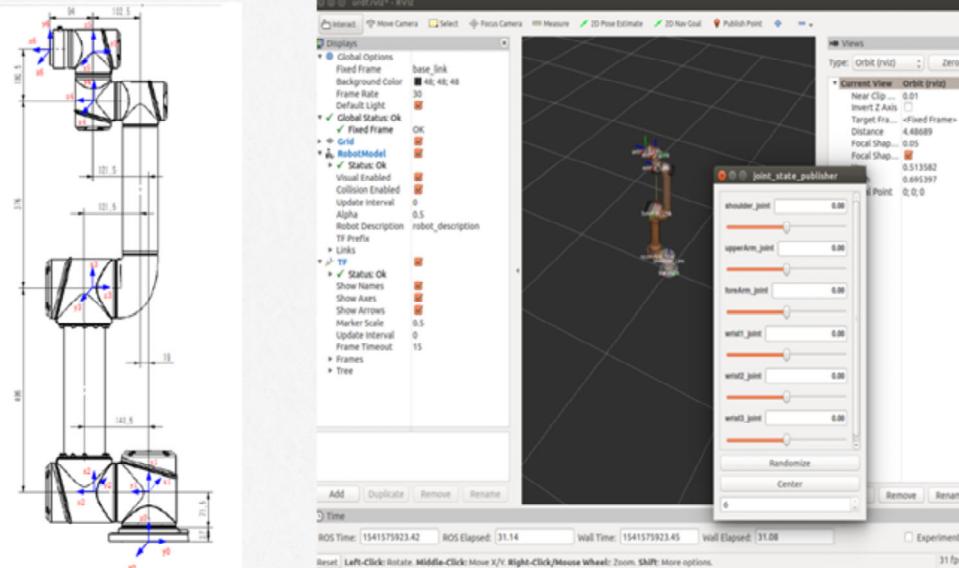


图2.23是ROS官方Wiki针对URDF的描述图，阐明了ROS中URDF相关的各个功能包(package)的关系。其中urdf, urdf\_parser\_plugin, collada\_parser都是robot\_model的一部分，目前已经从robot\_model中独立出来了。如果我们要采用URDF对机器人进行建模仿真，可以参照ROS中的源代码进行URDF的模型解析。

# 机器人建模描述文件与仿真

Robot Description and Simulation



<https://gitee.com/minhuasong/AUBO-Robot-on-ROS>

<https://github.com/mhuasong/AUBO-Robot-on-ROS>

# 机器人建模描述文件与仿真

Robot Description and Simulation

- URDF (Unified Robot Description Format, 统一机器人描述格式) -ROS
- SDF (Simulation Description Format) -Gazebo-着重于物理模拟
- MJCF (Multi-Joint Configuration Files /Contact Forces?)-多关节动力学仿真-MuJoCo

除了ROS起源的URDF之外，在Gazebo等仿真器中，基于URDF的发展，还有着重于物理模拟的SDF (Simulation Description Format) 描述文件格式。MJCF文件是MuJoCo中XML模型，MuJoCo更适合做多关节动力学仿真，它是一个物理引擎，旨在促进机器人技术、生物力学、图形和动画、机器学习以及需要快速准确模拟复杂动力系统领域的研究与开发。MJCF除了可以作为URDF扩展使用之外，还具有自己的格式以及一些更高级的模拟功能（例如肌腱等）。本章仅介绍了URDF的主要格式说明，详细的格式及用法请参见有关的软件说明。

# 07

## 机器人系统的设计方法

随着机器学习的发展，特别是在模块化、通用性、协作性以及人工智能等方面的发展，机器人技术有望从根本上改变人类文明在工业、农业、医疗、交通、通信等领域的运行方式。在设计机器人系统的时候，需要考虑系统性能、成本、可靠性、可维护性以及可扩展与升级等方面的因素。

# 机器人系统设计方法

Design Method of Robot System

## 设计原则

- 整体性原则。
- 通用性原则。
- 实物设计与仿真并行性原则

## 设计步骤

- 需求分析
- 概念设计
- 实施方案设计
- 详细设计
- 制造、安装、软硬件联调

**整体性原则。**机器人系统的设计包括机构、控制、传感等多方面的设计，在设计时需要首先考虑整体性原则。目前大多数的机器人还是为某种专业用途设计的，缺乏通用的机器人系统架构或模块化设计准则，解决此问题的一种解决方案是采用统一的系统架构，不同的机器人系统之间可以共享信息，提高系统软硬件的可重复使用性，并可对特定任务要求进行可扩展和可定制的调整。

**通用性原则。**选择通用的、可互换的组件进行设计。目前在机器人领域，最早出现标准化、可互换的零部件为驱动电机以及机器人专用减速器。采用标准化、可互换的零部件进行机器人系统的设计，不仅可以降低机器人系统设计的成本，也可以加速机器人系统的设计进度。除此之外，一些通用的组件库也可以大大降低机器人系统设计的难度，避免重复“造轮子”。

**实物设计与仿真并行性原则。**针对复杂的机器人系统，应尽量选择合适的仿真平台，在进行系统机构设计的同时，并行进行系统的建模仿真，在强大的物理仿真环境下对虚拟的机器人系统进行测试，确保虚拟系统和物理系统能够保持较好的一致性。

机器人系统的设计一般可以分成五个主要的步骤。

### 需求分析

需求分析阶段主要由设计人员与需求方协商分析机器人系统的要求，然后从功能及子功能的角度出发，根据系统的需求和资源，确定机器人采用的系统架构以及需要实现的性能参数，列出可以实现所有功能的功能载体，根据系统的约束条件，制定系统的设计规范。

机器人系统的功能指需要通过设计一定的对象来满足的一般任务。例如：螺丝紧固、目标分拣等功能。功能载体定义为可以实现某个功能的组件或组装件，例如可以实现螺丝紧固或目标分拣的机器人末端夹具。设计规范指设计机器人系统需要满足的定量条件，例如：搬运机器人的工作范围、搬运工件的重量以及流水线的节拍速度等约束条件。

### 概念设计

对机器人系统架构、性能参数进行分析，并确定了设计规范之后，在概念设计阶段，需要将机器人系统的功能分解至合适的细粒度，然后对于每一个子功能，找出多种实现手段或可替代方案，最后对这些对比方案进行彻底的分析和审查，并对候选方案进行时间进度与预算成本等方面的审查，直到确定一组合适方案作为最有可能满足所有功能和设计规范的候选解决方案。

### 实施方案设计

在实施方案设计阶段，一般以草图和初步设计图的形式给出概念设计阶段确定的备选方案，然后制定设计标准，以确定哪些替代方案最有可能满足需求。例如，在串联型机械臂的设计中，可以根据需求分析与概念设计之后确定机械臂自由度、关节类型、连杆长度、负载等参数模型，对机械臂的运动学，包括工作空间、奇异位形甚至动力学等进行仿真分析，生成所选方案的参数模型，并对该模型进行分析与优化设计。

### 详细设计

确定实施方案后，即可进行详细设计。在详细设计阶段，机器人控制系统的设计一般优先于机构设计。机器人控制系统的设计详见第6章的描述，控制系统设计主要包含控制方式的选择以及驱动方式选择。控制方式有集中式控制、分布式控制等方式。所谓集中式控制方式，即将所有资源集中在一个控制器上，在一个控制器上进行机器人整体的作业控制、运动控制以及所有关节的控制。而分布式控制方式，不同的控制器负责实现不同的功能，例如机器人的每个关节都有独立的控制器，运动控制器和作业控制器也可以不同，控制器之间通过总线（例如RS485、CAN、EtherCAT等总线）进行通信。目前较为成熟的机器人驱动方式主要有电动、液压、气动三种方式，机器人的驱动方式可根据机器人的作业环境、负载、体积等要求进行选择。一般来说液压驱动的负载最大、气动次之、电动最小。目前通用型的工业机器人，电驱动方式最为常用，又分为伺服电机、步进电机以及普通电机等，可根据电源类型，选择交流电机或直流电机。

机器人系统的功能，除了机构的限制之外，更多体现在控制系统的软件上面。机器人控制系统的软件开发可在虚拟仿真平台上并行进行。可在仿真平台软件上对机器人系统进行建模，并进行软件开发与仿真控制。

以工业机器人为例，机器人系统的机构设计包括底座、关节、连杆、末端执行器的设计，甚至还包括行走机构设计。机器人的机构设计较为复杂，需要设置专门的课程，本教材并不包含这部分。

在详细设计阶段，需要确定所有的零部件，这些零部件要么重新设计，要么从标准件目录中进行选型。在详细设计阶段，需要形成一份设计报告，其中应含：需求分析总结、备选的所有解决方案、选定解决方案的详尽描述、制造图、表以及预算估计等内容。

### 制造、安装、软硬件联调

在详细设计完成之后，可经过仿真测试，确认系统设计无误之后，即可进行

非标件的制造、标准件的选购，然后进行各子系统的调试、总体安装以及测试。在机器人系统的实物组装完成之后，可以进行软硬件的联调。

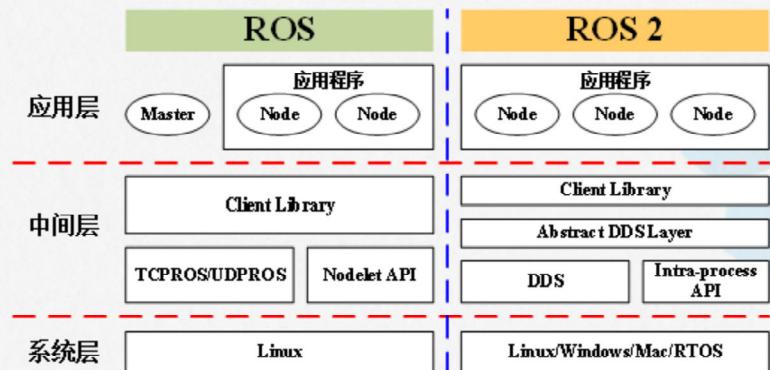
# 08

## 基于ROS的机器人系统设计

近几年，随着机器人操作系统ROS (Robot Operating System) 的普及，各种基于ROS的机器人系统、应用和软件成为主流。ROS平台使得机器人应用开发者无需关注机器人操作系统和底层硬件平台的具体实现细节，而把研究重点放到应用功能开发方面，从而极大加快了机器人系统的设计和开发的效率和水平。

# 基于ROS的机器人系统设计

Robot System Design On ROS



ROS是一个机器人框架，包含了大量开发工具、功能库代码和接口协议，旨在降低机器人开发这一过程的难度与复杂度。ROS设计者将ROS表述为“**ROS = Plumbing + Tools + Capabilities + Community**”，即ROS是通讯机制、工具软件包、机器人技能以及机器人社区的集合体，如图2.30所示。这四部分的特点如下。

**通信机制：**点对点的分布式通信机制是ROS的核心，节点间可以执行若干种类型的通信，包括基于话题（Topic）的异步数据流通信，基于服务（Service）的同步数据流通信，还有参数服务器上的数据存储等。

**工具软件包：**ROS提供了Rviz（三维可视化平台）、Qt工具箱、命令行工具、Gazebo（仿真软件）、MATLAB工具包以及大量的第三方工具。

**机器人技能：**ROS提供了机器人系统中常用硬件的ROS驱动包，如RGBD相机、激光雷达、机械臂、末端执行器等；提供了机器人上层功能包，如SLAM（即时定位与制图）、物体识别、路径规划、运动规划等；提供了一些便于开发的组件，如TF（Transformer）坐标系转换、Actionlib、Message等。

**机器人社区：**国内外常用的机器人平台大部分都已经支持ROS系统，并将相应的软件包发布到了ROS机器人社区，为机器人硬件设计参考、辅助机器人原型搭建、基于应用功能包的二次开发、各种算法验证以及机器人系统学习提供了便利。

ROS项目的初衷是为了给科研机器人Willow Garage PR2提供一个开发环境。但随着ROS开始广泛应用于各类智能系统，其原来的功能设计已经不能满足一些智能系统对于某些性能（如实时性、安全性、可移植性、多机器人协同等）的需求。为了满足各类机器人系统的需求，ROS开发者团队推出了ROS 2。

图2.31是ROS和ROS 2的系统架构对比。ROS2相对于ROS的改进主要体现在以下几点。

进程管理：在ROS中，需要开启中央节点管理器Master统一管理所有节点。如果Master节点出现故障，将严重影响ROS系统功能。而在ROS 2中，系统引入节点自发现机制，可有效提高系统鲁棒性。

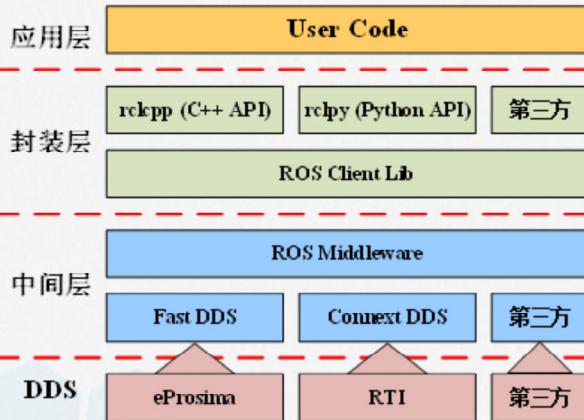
进程间通信：ROS基于TCP和UDP协议开发了TCPPROS和UDPROS协议。而在ROS 2中，通信协议更换成了更加复杂但也更加完善的**DDS**（Data-Distribution Service）系统。

进程内通信：对于在进程内进行大量数据的通信，ROS 和ROS 2都提供了基于共享内存的通信方法，分别是Nodelet和 Intra-process模块。

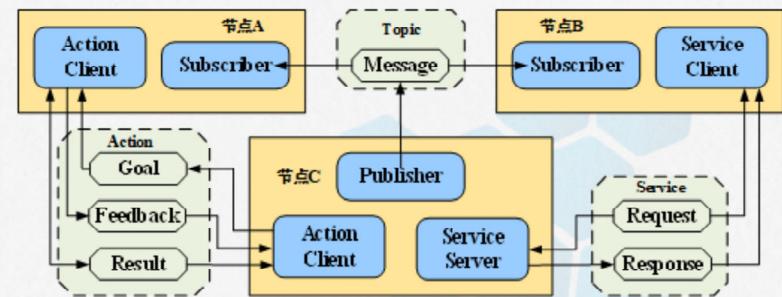
跨平台：ROS主要安装在Linux上，ROS 2可跨平台，如Linux、windows、MacOS、RTOS。

# 基于ROS的机器人系统设计

Robot System Design On ROS



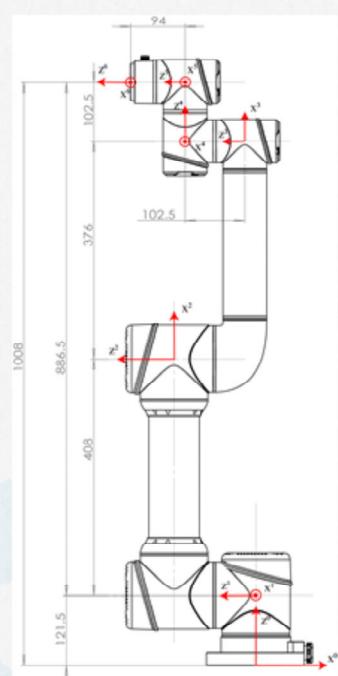
ROS 2 内部通信架构



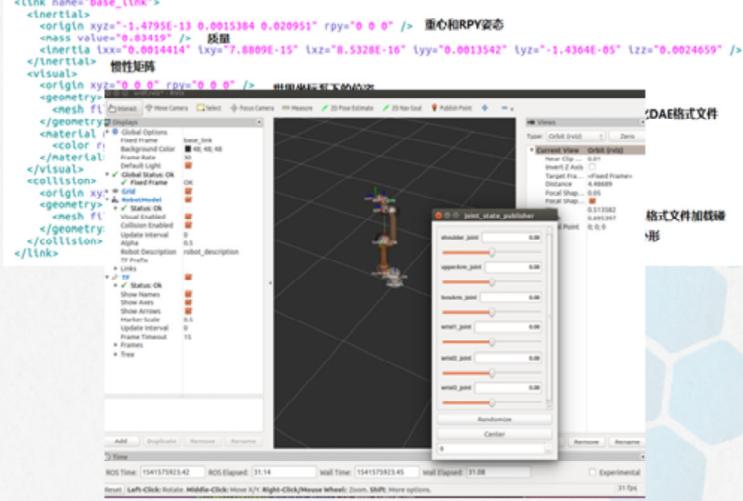
ROS 2 节点间的通信方式

# 基于ROS的机器人系统设计

Robot System Design On ROS



## 基于ROS的机械臂仿真设计实例 URDF

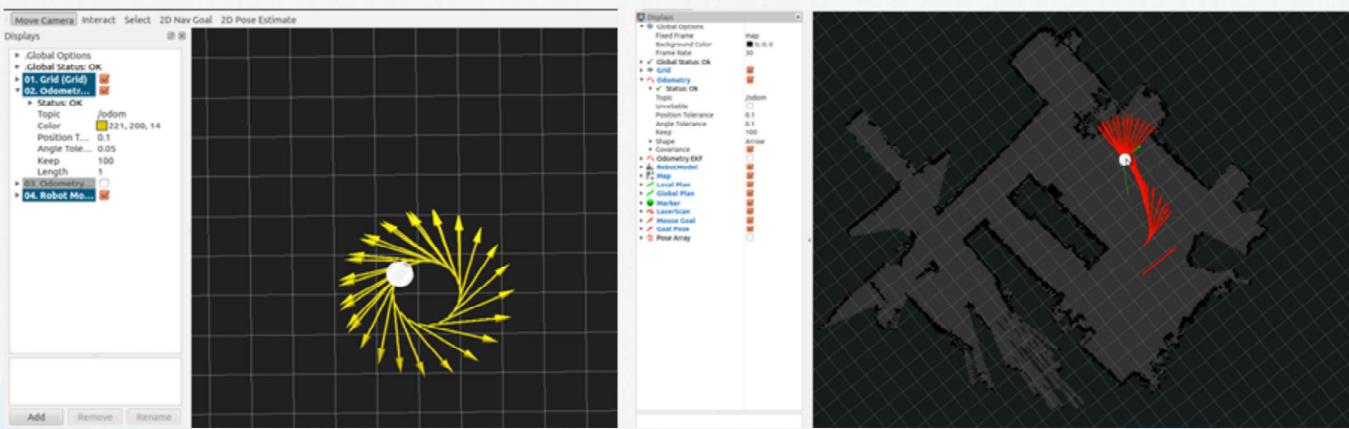


[https://github.com/mhuasong/Basics-of-Robotics-Theory-and-Technology/tree/main/ch2/ex2\\_7](https://github.com/mhuasong/Basics-of-Robotics-Theory-and-Technology/tree/main/ch2/ex2_7)

# 基于ROS的机器人系统设计

Robot System Design On ROS

## 基于ROS的移动机器人仿真设计实例

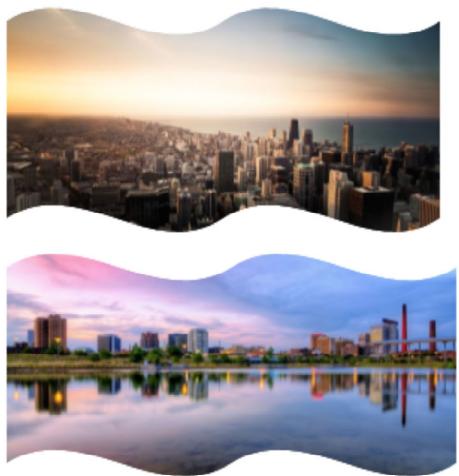


# 09

## 扩展阅读

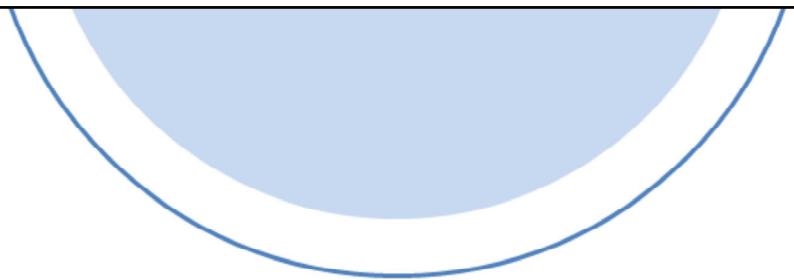
- SPA (Sense-Plan-Act) 范式
- Subsumption Architecture
- Hybrid Architecture
- PDDL(Planning Domain Description Language)
- ASP(Answer Set Programming)

机器人规划系统的发展趋势是实时地规划与执行，即机器人在作业的过程中通过描述模型和操作模型不断地规划、监视、修改和更新规划方案。因此越来越多的研究者结合描述模型和操作模型的特点来设计统一的规划模型，实现从高层推理到底层控制的统一表述，以便于机器人系统能够根据外部环境变化不断地修正规划方案，从而让机器人能够更好的适应环境变化。



## 课后练习

- ① 尝试编写不同体系架构的示范代码（可以考虑用C++或matlab的m语言进行编写，只需编写系统主题框架示范代码，并运行测试）。
- ② 编写RPY角法（滚动角、俯仰角、偏航角）、欧拉角法、罗德里格斯公式、四元数法以及罗德里格斯参数法相互之间的转换函数（C++/matlab），并测试验证。
- ③ 编写一个典型机器人的urdf文件，并在最新版的matlab、CoppeliaSim(V-rep)、ROS、MuJoCo下加载测试。



# THANK YOU

机器小学

Robotics