

# Table of Contents

关于本课程	1.1
开始之前	1.2
课程实施说明	1.2.1
课前读物	1.2.2
环境准备	1.2.3
第1章 基础知识	1.3
第1节 Python语言基础	1.3.1
第2节 Linux基本操作	1.3.2
第3节 Web服务基础	1.3.3
第4节 开源硬件基本操作	1.3.4
第5节 认识硬件—树莓派	1.3.5
第6节 认识硬件—Arduino	1.3.6
第7节 认识硬件—ESP8266	1.3.7
第8节 嵌入式系统软件开发流程	1.3.8
第2章 人工智能初体验	1.4
第1节 人工智能、机器学习的相关概念	1.4.1
第2节 知识图谱	1.4.2
第3节 前馈神经网络（FFNN）	1.4.3
第4节 卷积神经网络（CNN）	1.4.4
第5节 循环神经网络（RNN）	1.4.5
第6节 人工智能与游戏（DQN）	1.4.6
第3章 硬件基础：智能小白	1.5
第1节 小白的心脏：esp8266开发板	1.5.1
第2节 无线控制：遥控小车	1.5.2
第3节 汽车人小白：机械臂	1.5.3
第4节 视物而行：远程视频救援	1.5.4
第4章 机器视觉：自动追踪小车大白	1.6
第1节 环境准备：借我一双慧眼吧	1.6.1
第2节 大白智能分拣	1.6.2
第3节 大白自动追踪	1.6.3
第5章 深度学习：无人驾驶小车老白	1.7
第1节 环境准备：更加专业的视觉系统	1.7.1
第2节 无人驾驶数据采集、训练与测试	1.7.2
第3节 红灯停绿灯行：识别交通信号	1.7.3
第6章 综合进阶：机器人小绿	1.8

第1节 环境准备：物联网平台安装配置	1.8.1
第2节 小绿的一小步，我们的一大步	1.8.2
第3节 “小绿，跳舞！”——制作自己的语音助手	1.8.3
第4节 使用物联网制作人脸解锁	1.8.4
第5节 物联网进阶：执行自动化操作	1.8.5
第6节 唱跳rap：小绿实时姿态模仿	1.8.6
资料下载	1.9

# 人工智能课程介绍

## 概述

- 本项目包含一个[在线文档](#)以及配套的教学材料（PPT、教案、学习单）。
- 包含6个有所关联、逐步递进的项目构成的章节。分别涉及物联网、机器人、机器视觉、语音识别和控制等内容。
- 包含背景知识学习、环境准备和资源下载部分。

本课程通过6个有所关联、逐步递进的章节，探究人工智能的基本原理，了解人工智能、机器学习基本概念，参与计算机视觉、语音技术等人工智能领域相关项目，利用人工智能解决学习和生活中的实际问题。

课程涉及人工智能基本概念、计算机视觉、语音技术、机器人等内容。通过一系列动手实验，制作小车和机器人，完成物体检测、自动追踪、无人驾驶、机器人姿态模仿、语音助手等项目。

课程包含详细的在线操作文档、配套代码、小车及机器人3D打印零件及全部硬件器材，学生无需自行准备任何设备，即可顺利完成全部项目。

完成课程学习后，将会对人工智能的基本概念有一定了解；对图像分类，目标检测，自然语言处理、语音识别等技术有直观体会；掌握开源硬件的基本操作，并通过开源硬件构建人工智能应用。

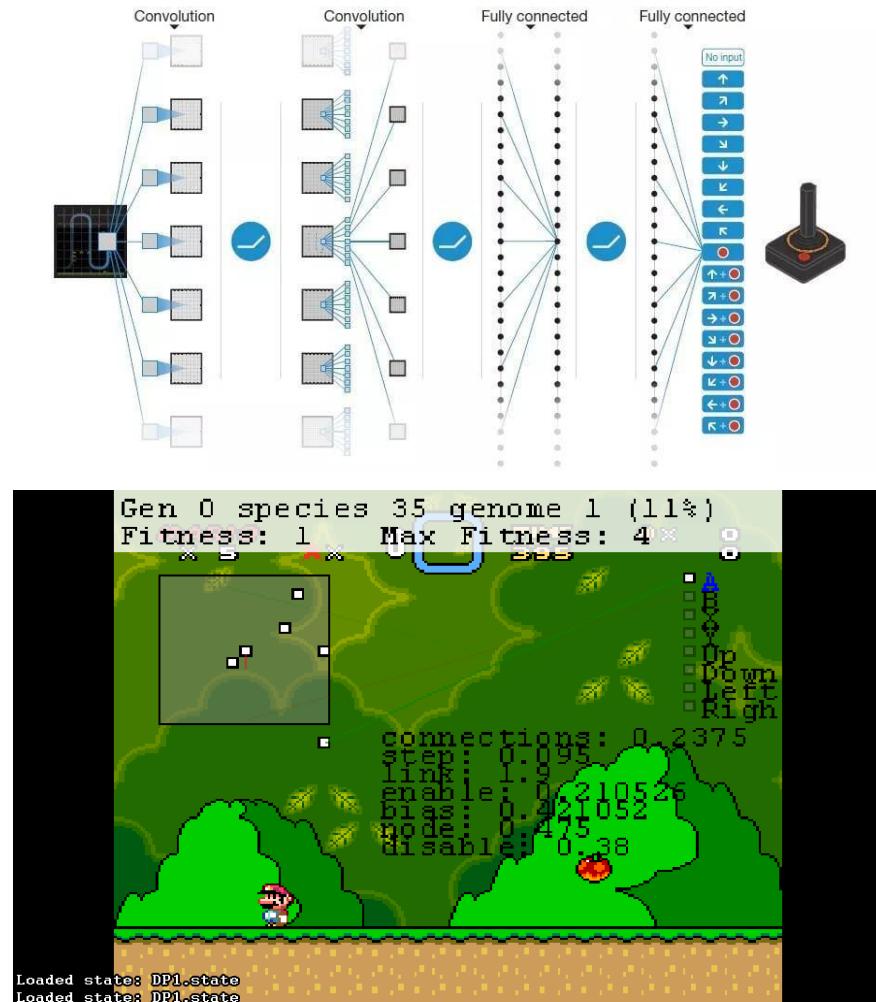
## 分章节介绍

### 第1章 基础知识

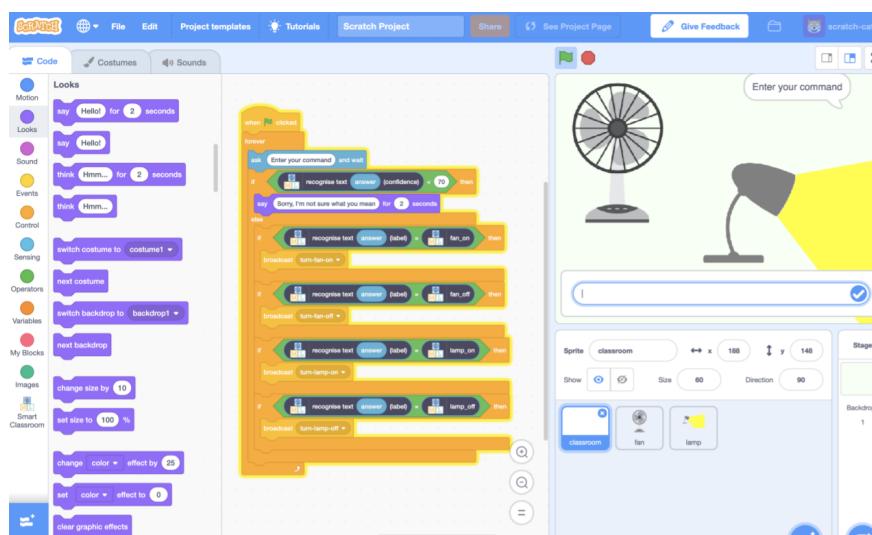
- 介绍：

### 第2章 人工智能体验

- 介绍： 通过本章内容，学生对人工智能的数学基础、概率论和博弈论、人工智能在图形图像和语言处理、电子游戏及其他领域的应用有感性认识。



使用在线的积木编程（google blockly/scratch）来控制





- 涉及软硬件： PC或树莓派（Raspberry Pi）或NVIDIA Jetson Nano

## 第3章 智能车“小白”

- 介绍： 主要使用ESP8266提供Web服务，来控制电机、舵机、传感器等，实现远程遥控、远程视频监控、巡线、避障、控制机械臂抓取等功能 可以通过积木编程的方式来对小车的功能进行编程

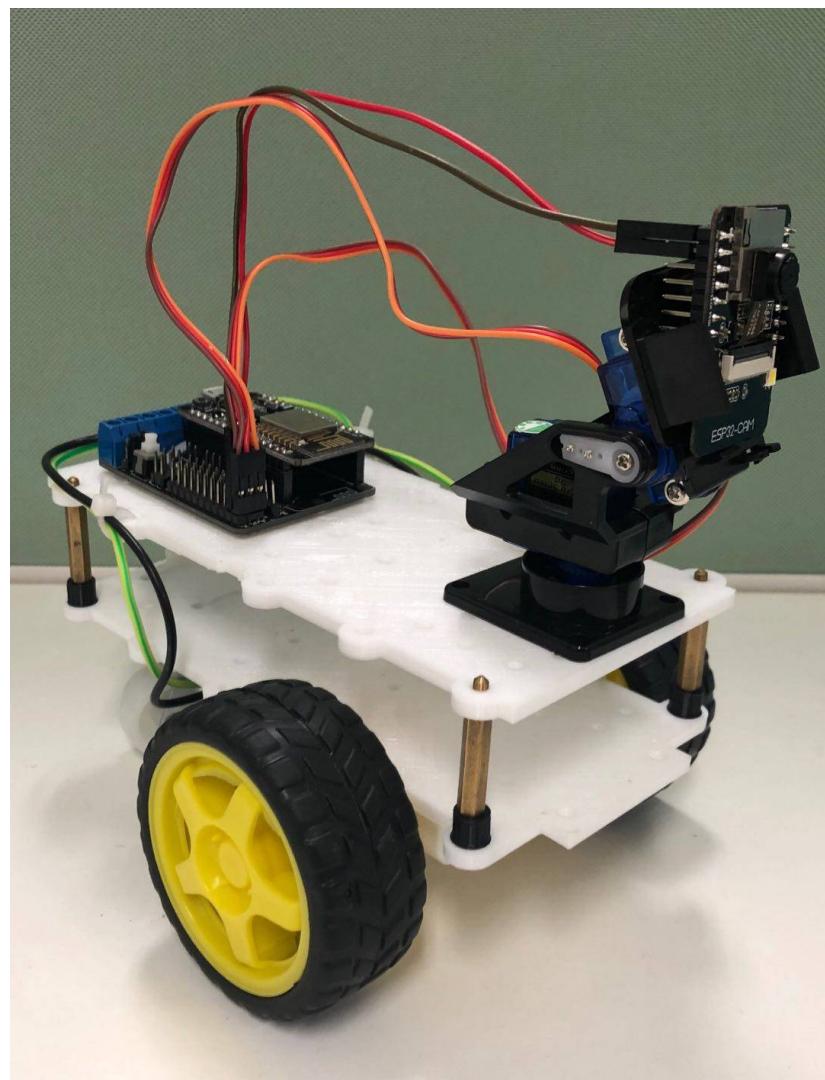
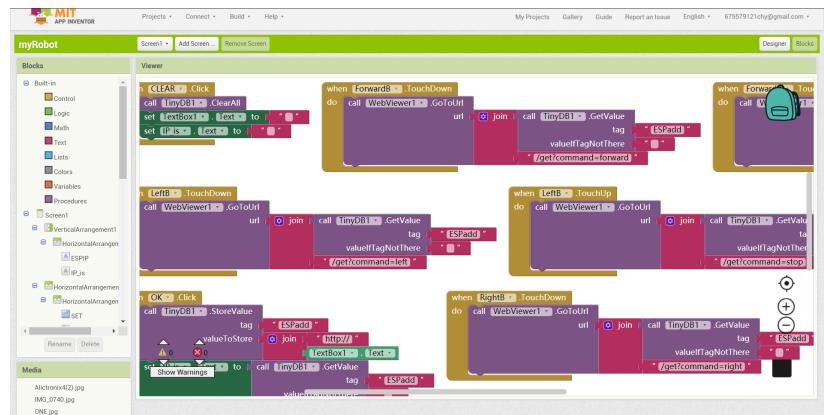
The top screenshot shows a browser-based Scratch-like editor. The script consists of the following blocks:

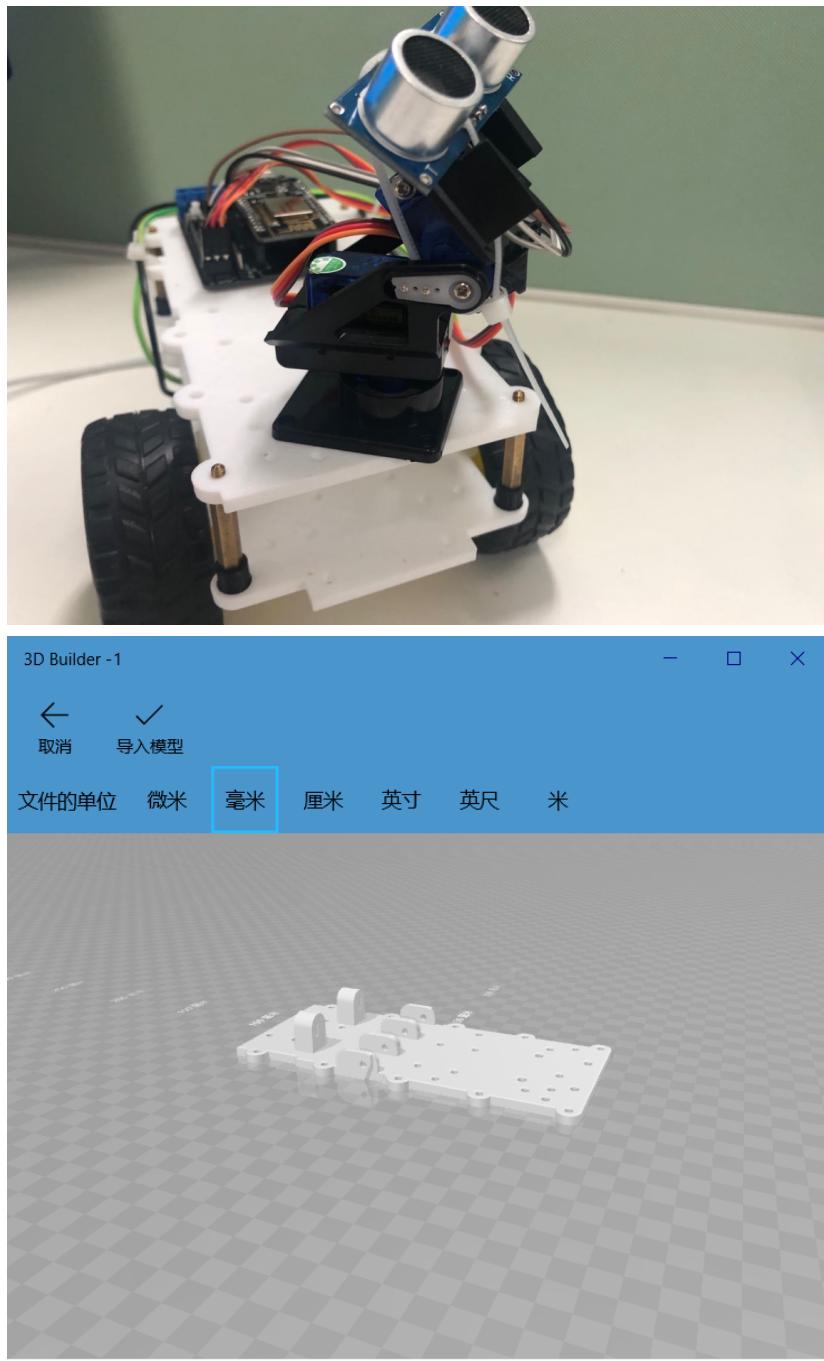
```

1- if sumorobot.is_opponent():
2   sumorobot.set_led(STATUS, True)
3   sumorobot.move(STOP)
4   sumorobot.set_servo(LEFT, 100)
5   sumorobot.move(FORWARD)
6- else:
7   sumorobot.move(SEARCH)

```

The bottom screenshot shows the MIT App Inventor interface for creating a mobile application named "myRobot". The screen design includes a 3x3 grid of buttons labeled with arrows pointing up, down, left, and right. The properties panel on the right shows settings for the screen component.



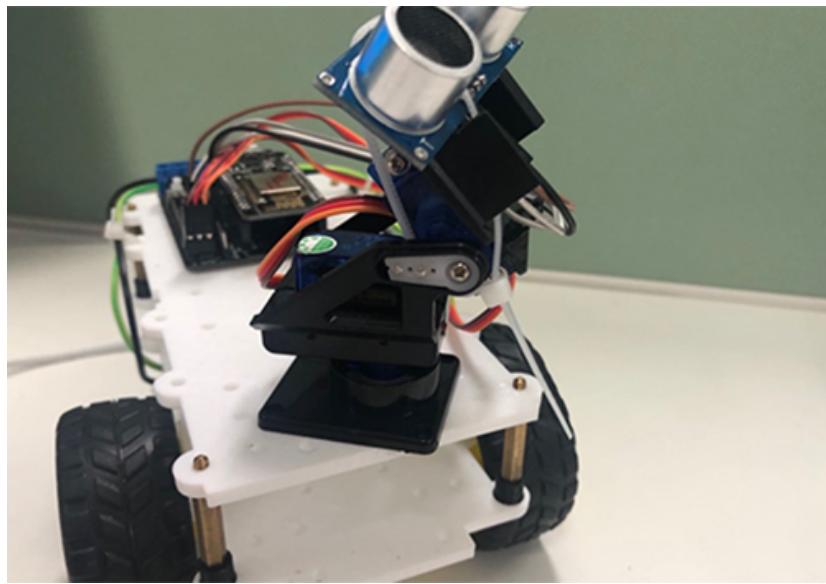


- 涉及软硬件: ESP8266, ESP32-CAM, 小车套件 (可3D打印), 舵机, 灰度传感器、超声波传感器等 自行开发的物联网控制平台, MIT App Inventor

## 第4章 自动追踪小车“大白”

- 介绍: 从机器视觉出发, 让学生理解机器视觉的相关概念和原理, 辨别 OpenCV和深度学习的异同点。使用OpenCV来处理视觉信号, 并通过蓝牙或串口来将处理过的视觉信号发送给小车, 从而实现物体追踪, 人脸追踪, 智能机械臂抓取等功能 学生通过使用Python, 完成信息采集: 爬虫、多文件处理; 信息处理: 训练采集的数据, 形成分类器, 从而让计算机视觉系统能够对

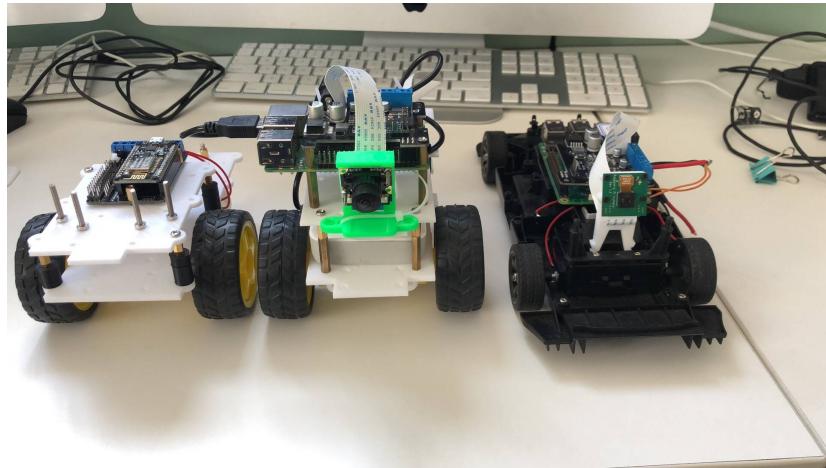
特定的物体进行分辨



- 涉及软硬件：树莓派、Arduino、舵机、USB摄像头、小车套件、3D打印机、电磁传感器、蓝牙接收器 OpenCV、Python

## 第5章 无人驾驶小车“老白”

- 介绍：采用深度学习的方式，通过采集无人驾驶的数据，并进行训练，来实现无人驾驶的功能

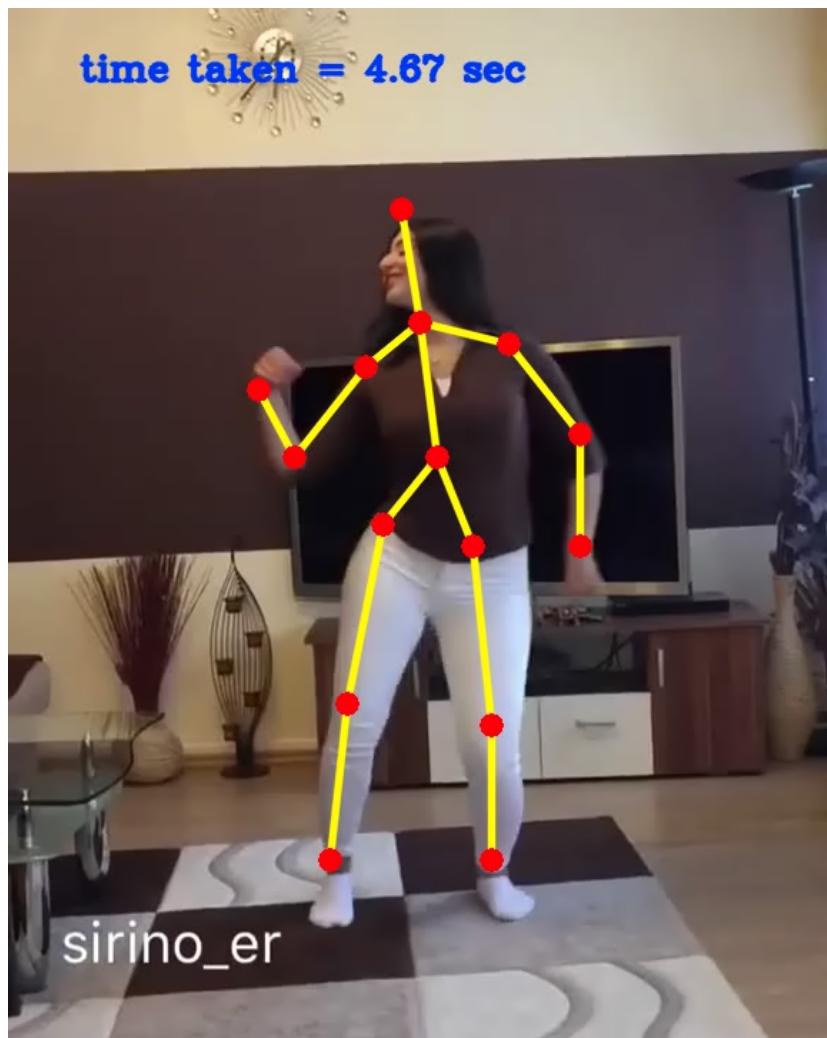




- 涉及软硬件： 树莓派、摄像头、小车套件 Python

## 第6章 机器人“小绿”

- 介绍： 组装一个机器人，作为物联网的一个节点，实现多种物联网功能，包括  
网页遥控：通过自行开发的物联网平台来对它进行遥控；语音助手：可以通过自己训练的热词来进行唤醒、通过语音来控制机器人执行各种动作；控制其他设备：比如控制前几个章节的小车，读取各种传感器的数据等；人脸解锁：通过实时的人脸识别和红外线发射装置，实现人脸解锁，也可以通过Google Assistant、Siri、Alexa等远程控制；实时姿态模仿：通过单目摄像头拍摄实时画面，采用OpenPose姿态识别软件进行处理，将关节姿态数据通过蓝牙或串口传递给机器人，机器人进行实时的姿态模仿。



- 涉及软硬件：树莓派、ESP8266、麦克风阵列、舵机、3D打印机、摄像头等

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间：2020-02-26

# 开始之前

---

本部分为本课程之基础和说明

## 课程实施说明

介绍了课程的目标、大纲、预备要求、学分授予和参考资源

## 课前读物

部分旨在让同学们在学习之前对计算机和人工智能的相关概念、应用有一定的了解，对课程涉及到的硬件进行了简单的介绍

## 环境准备

部分详细说明了课程实施的全程中需要的软硬件环境

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-02-26

# 课程实施说明

## 课程目标

- 了解计算机基本知识
- 了解人工智能、机器学习基本概念
- 体验人工智能在实际中的应用，会使用开源硬件解决实际问题

## 课程大纲

章节	主题	时间分配
前导	基础入门	1课时
第1章	应用体验	2课时
第2章	物联网	? 课时
第3章	熟悉硬件	4课时
第4章	机器视觉	4课时
第5章	无人驾驶	4课时
第6章	综合进阶	4课时

## 预备要求

- 掌握计算机基本操作
- 对编程语言有简单的了解
- 少量内容需要一定的阅读能力

## 学分授予

本课程根据学生学习的总评成绩发放学习证书并认定学分，总评成绩超过60分给予课程学分，此外总评成绩在60-84分发放合格证书，总评成绩大于84分发放优秀证书。总评成绩的计算公式如下：

$$\text{总评成绩} = \text{平时} * 30\% + \text{出勤率} * 20\% + \text{期末考试成绩} * 50\%$$

其中： 平时成绩：每个章节测试成绩满分为100分，7个章节总成绩取平均值作为平时成绩，按照比例计入总分； 出勤率：按照出勤情况给予打分，全勤则为满分，按照比例计入总分； 期末考试成绩：满分100分，考试试题为实践操作，教师当场打分，按照比例计入总分。

## 参考资源

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook修订时间：2019-12-15

# 课前读物

---

## 计算机概述

### 计算机的发展

#### 第一代：真空管

##### **ENICA**

**ENICA**（电子数值积分计算机，*Electronic Numerical Integrator And Computer*）是世界上第一台通用计算机。它是图灵完全的电子计算机，能够重新编程，解决各种计算问题。

**ENIAC**为美国陆军的弹道研究实验室（*Ballistic Research Laboratory, BRL*）所使用，用于计算火炮的外弹道。美国军方要求该实验室每天为陆军炮弹部队提供6张射表以便对导弹的研制进行技术鉴定。事实上每张射表都要计算几百条弹道，而每条弹道的数学模型是一组非常复杂的非线性方程组。这些方程组是没有办法求出准确解的，因此只能用数值方法近似地进行计算。按当时的计算工具，实验室即使雇用200多名计算员加班加点工作也大约需要二个多月的时间才能算完一张射表。

当时任职宾夕法尼亚大学莫尔电机工程学院的约翰·莫奇利（John Mauchly）和他的研究生约翰·埃卡特（John Eckert）提出了使用真空管制造电子计算机的设想。1934年，代号**PX**项目的计算机研制工作在宾夕法尼亚大学的电气工程摩尔商学院秘密展开。但由于**ENICA**使用真空管制作，使得**ENICA**计算机十分巨大，重达30吨，占地1500平方英尺，包含超过18000个真空管，在运行时需要消耗140千瓦的功率。同时它也比任何机电计算机快得多，每秒可以计算5000次加法运算。**ENIAC**可以在30秒内计算出人类计算20小时的导弹轨迹。

**ENIAC**于1946年完成，已无法用于战争。相反，它的首要任务是执行一系列复杂的计算，用于帮助确定氢弹的可行性。**ENIAC**用于其他目的之外的目的证明了其通用性。**ENIAC**继续在弹道研究实验室（*BRL*）管理下运营，直到1955年被拆解。

##### 冯·诺伊曼结构

启动和改变**ENIAC**程序的任务非常繁琐。但是假设程序可以以适合与数据一起存储在存储器中的形式表示。然后，计算机可以通过从存储器中读取它们来获得其指令，并且可以通过设置一部分存储器的值来设置或改变程序。

这个被称为存储程序概念的想法通常归功于**ENIAC**设计师，最著名的是数学家冯·诺依曼（John von Neumann），他是**ENIAC**项目的顾问。阿兰·图灵（Alan Mathison Turing）几乎同时发明了这个想法。这个想法首次发表于冯·诺依曼在1945年提出的一种新计算机**EDVAC**（电子离散可变计算机）的提案。

1946年，冯·诺伊曼和他的同事开始在普林斯顿高等研究院开设一种新的存储程序计算机，称为**IAS**计算机。**IAS**计算机虽然直到1952年才完成，但它是所有后续通用计算机的原型。

## 第二代：晶体管

电子计算机的第一个重大变化是用晶体管替换真空管。与真空管相比，晶体管更小，更便宜并且散热更少，但是可以以与真空管相同的方式用于构造计算机。与需要电线，金属板，玻璃胶囊和真空的真空管不同，晶体管是由硅制成的固态器件。

该晶体管于1947年在贝尔实验室被发明，并在20世纪50年代引发了电子革命。然而，直到20世纪50年代后期，完全晶体管化的计算机才被商业化。随后IBM推出了著名的7000系列计算机。

晶体管的使用定义了第二代计算机。基于所采用的基本硬件技术，将计算机分类为几代已被广泛接受。每一代新产品的特点是处理性能更高，内存容量更大，并且尺寸比前一代更小。

计算机代	时间	技术	速度（每秒计算次数）
1	1946-1957	真空管	40,000
2	1958-1964	晶体管	200,000
3	1965-1971	小规模和中等规模集成电路	1,000,000
4	1972-1977	大规模集成电路	10,000,000
5	1978-1991	超大规模集成电路	100,000,000
6	1991-	特大規模集成电路	1,000,000,000

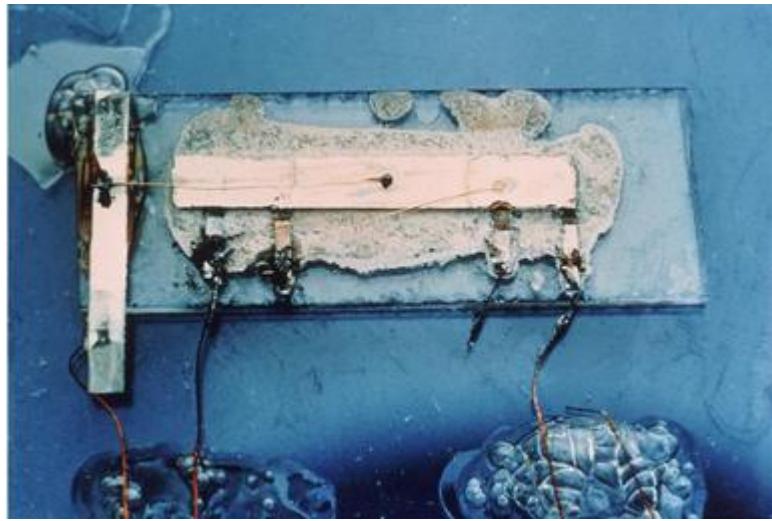
除此之外，第二代计算机在结构上也有了一些其他变化。第二代计算机引入了更复杂的算术和逻辑单元和控制单元，使用高级编程语言，以及与计算机一起提供系统软件。从广义上讲，系统软件提供了加载程序，将数据移动到外围设备和库以执行常见计算的能力，类似Windows和Linux等现代操作系统。

第二代也值得注意的是数字设备公司（DEC）的出现。DEC成立于1957年，并在那一年交付了第一台计算机PDP-1。

## 第三代：集成电路

单个独立的晶体管称为分立元件。在20世纪50年代和60年代早期，电子设备主要由分立元件组成——晶体管，电阻器，电容器等。分立元件分别制造，封装在自己的容器中，焊接或连接在一起形成类似绝缘纤维板的电路板，然后安装在计算机、示波器和其他电子设备中。每当一个电子设备需要一个晶体管时，一个含有针头大小的硅片的小管子必须焊接到电路板上。从晶体管到电路板的整个制造过程既昂贵又麻烦。

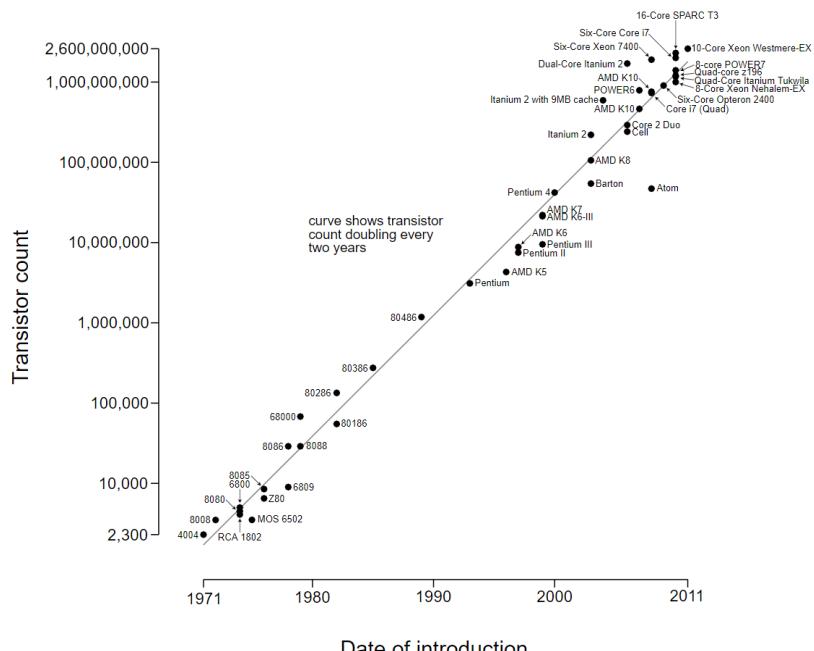
早期的第二代计算机包含大约10000个晶体管。这个数字增长到数十万，使得制造更新，更强大的机器变得越来越困难。



1958年取得了革命性的电子技术并开创了微电子时代的成就：集成电路的发明。它是定义第三代计算机的集成电路。集成电路利用了诸如晶体管，电阻器和导体之类的部件可以由诸如硅的半导体制造的事实。它仅仅是固态技术的延伸，用于在一小块硅中制造整个电路，而不是将由单独的硅片制成的分立元件组装到同一电路中。许多晶体管可以在单个硅晶片上同时生产。同样重要的是，这些晶体管可以通过金属化工艺连接以形成电路。

最初，只有少数门或存储单元可以可靠地制造和封装在一起。这些早期的集成电路称为小规模集成电路（Small Scale Integration, SSI）。随着时间的推移，可以在同一芯片上打包越来越多的组件，晶体管的密度的增长如图所示。

Microprocessor transistor counts 1971-2011 & Moore's law



这个数字也反映了1965年英特尔联合创始人戈登·摩尔（Gordon Moore）提出的著名的摩尔定律。摩尔观察到可以放在单个芯片上的晶体管数量每年翻倍并预测这种速度会继续到不久的将来。令包括摩尔在内的许多人惊讶的是，这种速度一年又

一年，十年后持续不断。在 20世纪70年代，节奏减缓到每18个月翻一番，但此后一直保持这一速度。

### 摩尔定律

摩尔定律（**Moore's law**）是由英特尔创始人之一戈登·摩尔（**Gordon Moore**）提出的。其内容为：集成电路上可容纳的晶体管数目，约每隔两年便会增加一倍。经常被引用的**18个月**，是由英特尔首席执行官大卫·豪斯（**David House**）提出：预计**18个月**会将芯片的性能提高一倍（即更多的晶体管使其更快），是一种以倍数增长的观测。

尽管摩尔定律的现象已经被观察到了数十年，摩尔定律仍应该被视为是对现象的观测或对未来的推测，而不是一个物理定律或自然界的规律。从另一角度看，未来的增长率在逻辑上无法保证会跟过去的数据一样，也就是逻辑上无法保证摩尔定律会持续下去。虽然预计摩尔定律将持续到至少**2020年**，然而**2010年**国际半导体技术发展路线图的更新增长已经在**2013年年底**放缓；又比如说英特尔在**22纳米**跟**14纳米**的**CPU制程**上已经放慢了技术更新的脚步。

## 之后的几代

在第三代之后，关于定义几代计算机的一致意见较少。随着大规模集成（**Large Scale Integration, LSI**）的引入，可以在单个集成电路芯片上放置**1000**多个元件。  
超大规模集成（**Very Large Scale Integration, VLSI**）每个芯片实现了**10000**多个组件，而目前的超大规模集成（**Ultra Large Scale Integration, ULSI**）芯片可以包含超过**10亿**个组件。

### 半导体储存器

集成电路技术首次应用于计算机是集成电路芯片中处理器（控制单元和算术和逻辑单元）的构造，但人们也发现这种技术可用于构建储存器。

在**20世纪50年代**和**60年代**，大多数计算机储存器由铁磁材料的微小环构成，每个直径约为十六分之一英寸。这些环被悬挂在计算机内部小屏幕上的细线网格上。单向磁化，一个环（称为核心）代表**1**，另一方向磁化，它代表**0**。磁芯储存器相当快，读取存储在储存器中的一点只花了百万分之一秒，但它昂贵，笨重，并且使用了破坏性读数——读取核心的简单行为会删除存储在其中的数据。因此，需要安装额外的电路以便在提取数据后立即恢复数据。

然后，在**1970年**，仙童半导体（**Fairchild**）产生了第一个半导体储存器。这个芯片大小只有一个内核，可以容纳 **256** 位数据，访问速度比核心储存器快得多，只需要**700** 亿分之一秒的读取时间。

**1974年**，发生了一个重大事件：半导体储存器的每比特价格降至核心储存器的每比特价格之下。在此之后，储存器成本持续快速下降，伴随着物理存储密度的相应增加。这导致了更小、更快的机器，其内存尺寸更大，更昂贵的机器从几年前开始。内存技术的发展以及接下来要讨论的处理器技术的发展，在不到十年的时间里改变了计算机的性质。虽然笨重，昂贵的计算机仍然是这一领域的一部分，但计算机也已经被用于办公机器和个人计算机的最终用户。

### 微处理器

正如存储器芯片上的元件密度持续上升一样，处理器芯片上的元件密度也在不断增加。随着时间的推移，越来越多的元件被放置在每个芯片上，因此构建单个计算机处理器所需的芯片越来越少。

1971年英特尔开发出4004时取得了突破性进展。4004是第一款在单芯片上集成CPU所有组件的芯片——微处理器诞生了。

1972年英特尔推出了8008处理器。这是第一个8位微处理器，性能几乎是4004的两倍。

1974年英特尔8080的推出。这是第一个通用微处理器，与4004和8008是为特定应用而设计的不同，8080被设计成通用微型计算机的CPU。与8008一样，8080是一个8位微处理器。然而，8080更快，具有更丰富的指令集，并具有大的寻址能力。

同时16位微处理器的开发开始了。但是直到20世纪70年代末才出现了功能强大的通用16位微处理器，其中之一是8086。微处理器的下一次重大升级是发生在1981年，贝尔实验室和惠普公司都开发了32位单芯片微处理器。英特尔于1985年推出了自己的32位微处理器80386。

## 计算机的操作系统

操作系统是管理计算机硬件的程序，它还为应用程序提供基础，并且充当计算机硬件和计算机用户的中介。令人惊奇的是操作系统完成这些任务的方式多种多样。大型机的操作系统设计的主要目的是为了充分优化硬件的使用率，个人计算机的操作系统是为了能支持从复杂游戏到商业应用的各种事物，手持计算机的操作系统是为了给用户提供一个可以与计算机方便地交互并执行程序的环境。因此，有的操作系统设计是为了方便，有的设计是为了高效，而有的设计目标则是兼而有之。

## 操作系统做了什么

操作系统是几乎所有计算机系统的一个重要部分。计算机系统可以大致分为4个组成部分：计算机硬件、操作系统、系统程序与应用程序和用户。

硬件，如中央处理单元（Central Processing Unit, CPU）、内存（memory）输入输出设备（input/output devices, I/O devices），为系统提供基本的计算资源。应用程序如字处理程序、电子制表软件、编译器、网络浏览器规定了用户按何种方式使用这些资源。操作系统控制和协调各用户的应用程序对硬件的使用。

计算机系统的组成部分包括硬件、软件及数据。在计算机系统的操作过程中，操作系统提供了正确使用这些资源的方法。操作系统类似于政府。与政府一样，操作系统本身并不能实现任何有用的功能。它只不过提供了一个方便其他程序做有用工作的环境。

为了更加全面地理解操作系统所担当的角色，接下来从两个视角探索操作系统：即从用户的视角和系统的视角来讨论。

### 用户视角

计算机的用户观点因所使用接口的不同而异。绝大多数计算机用户坐在一台这样的PC前，PC由显示器、键盘、鼠标和主机组成。这类系统设计是为了让单个用户单独使用其资源，其目的是优化用户所进行的工作（或游戏）。对于这种情况，操作

系统的设计目的是为了用户使用方便，性能是次要的，而且不在乎资源使用率——如何共享硬件和软件资源。性能对用户来说非常重要，而不是资源使用率，这种系统主要为了优化单用户的情况。

在某些情况下，有些用户坐在与大型机或小型机相连的终端前，其他用户通过其他的终端访问同一计算机。这些用户共享资源并可交换信息。操作系统设计为资源使用做了优化：确保所有的CPU时间、内存和I/O都能得到充分使用，并且确保没有用户使用超出其权限以外的资源。

在另一些情况下，其他用户坐在工作站前，工作站与其他工作站和服务器相连。这些用户不但可以使用专用的资源，而且可以使用共享资源，如网络和服务器及文件、计算和打印服务器。因此，这类操作系统的设计目的是个人使用性能和资源利用率的折中。

近来，各种手持计算机开始成为时尚。绝大多数这些设备为单个用户所独立使用。有的也通过有线或（更为常见）无线与网络相连。由于受电源、速度和接口所限，它们只能执行相对较少的远程操作。绝大多数这类操作系统的功能是为了方便个人使用，当然如何在有限的电池容量中发挥最大的效用也很重要。

有的计算机几乎没有或根本没有用户观点。例如，在家电和汽车中所使用的嵌入式计算机可能只有键盘，只能打开和关闭指示灯来显示状态，而且这些设备及其操作系统通常设计成无需用户干预就能自行运行。

### 系统视角

从计算机的角度来看，操作系统是与硬件最为密切的程序，可以将操作系统看作资源分配器。计算机系统可能有许多资源，用来解决CPU时间、内存空间、文件存储空间、I/O设备等问题。操作系统管理这些资源。面对许多甚至冲突的资源请求，操作系统必须决定如何为各个程序和用户分配资源以便计算机系统能有效而公平地运行。众所周知，资源分配对多用户访问主机或微型计算机特别重要。

操作系统的一个稍稍不同的观点是强调控制各种设备和用户程序的需要。操作系统是控制程序。控制程序管理用户的执行以防止计算机资源的错误使用或使用不当。它特别关注I/O设备的操作和控制。

## 计算机语言与程序设计

编程语言（英语：Programming Language），是用来定义计算机程序的形式语言。它是一种被标准化的交流技巧，用来向计算机发出指令。一种计算机语言让程序员能够准确地定义计算机所需要使用的数据，并精确地定义在不同情况下所应当采取的行动。

最早的编程语言是在计算机发明之前产生的，当时是用来控制提花织布机及自动演奏钢琴的动作。在计算机领域已发明了上千不同的编程语言，而且每年仍有新的编程语言诞生。很多编程语言需要用指令方式说明计算的程序，而有些编程语言则属于宣告式编程，说明需要的结果，而不说明如何计算。

编程语言的描述一般可以分为语法及语义。语法是说明编程语言中，哪些符号或文字的组合方式是正确的，语义则是对于编程的解释。有些语言是用规格文件定义，例如C语言的规格文件也是ISO标准中一部分，而其他语言（像Perl）有一份主要的编程语言实现文件，视为是参考实现。

## 历史

## 早期发展

非常早期的计算机，例如巨人计算机（Colossus），在没有存储程序的帮助下，通过修改其电路或设置物理控制组来编程。

稍后，程序可以用机器语言编写，程序员以硬件可以直接执行的数字形式写入每条指令。例如，在两个存储器位置添加值的指令可能包含3个数字：一个选择“添加”操作的“操作码”和两个存储器位置。这些程序以十进制或二进制形式从穿孔卡，纸带，磁带读取或在计算机前面板上的开关上切换。机器语言后来被称为第一代编程语言。

下一步是开发第二代编程语言或汇编语言，它们仍然与特定计算机的指令集架构密切相关。这些使得程序更具人性化，并使程序员免于繁琐且容易出错的地址计算。

第一代高级编程语言或第三代编程语言是在20世纪50年代编写的。为计算机设计的早期高级编程语言是Plankalkül，由康拉德·楚泽（Konrad Zuse）在1943年至1945年间为德国Z3计算机开发。然而，它直到1998年和2000年才编写完成。

约翰·莫奇利（John Mauchly）在1949年提出短代码编程语言（Short Code），是有史以来为电子计算机开发的第一批高级语言之一。与机器代码不同，短代码语句以可理解的形式表示数学表达式。但是，程序每次运行时都必须转换为机器代码，这使得该过程比运行等效的机器代码慢得多。

在曼彻斯特大学，艾里克·格伦尼（Alick Glennie）在20世纪50年代初开发了Autocode。它使用编译器自动将语言转换为机器代码。第一个代码和编译器是在1952年为曼彻斯特大学的Mark 1计算机开发的，被认为是第一个编译的高级编程语言。

第二个Autocode是由托尼·布鲁克（RA Brooker）于1954年为Mark 1开发的，被称为Mark 1 Autocode。布鲁克还在20世纪50年代与曼彻斯特大学合作开发了Ferranti Mercury（一种计算机）的自动编码。这种语言的EDSAC计算机版本在1961年被剑桥大学数学实验室的哈特利（David Hartley）设计，被称为EDSAC 2 Autocode。它是从Mercury Autocode的基础上根据新机器的需求修改而来，可以指出其目标代码的优化和源语言当时推进的诊断。

1954年，FORTRAN语言由布鲁克（John Backus）在IBM发明。它是第一个广泛使用的高级通用编程语言，具有功能实现，而不仅仅是纸上设计。它仍然是高性能计算的流行语言，用于对世界上最快的超级计算机进行基准测试和排名的程序。

另一种早期编程语言是由格蕾丝·赫柏（Grace Hopper）在美国设计的，名为FLOW-MATIC。它是最初是在1955年至1959年期间由雷明顿兰德公司（Remington Rand）为UNIVAC I计算机开发的。赫柏发现商业数据处理客户对数学符号感到不舒服，并且在1955年初，她和她的团队编写了英语编程语言规范并实施原型。FLOW-MATIC编译器于1958年初公开上市，并于1959年基本完成。

## 细化

越来越多的高级语言的使用引入了对低级编程语言或系统编程语言的要求。这些语言在不同程度上提供汇编语言和高级语言之间的便利，它们可用于执行需要直接访问硬件设施但仍提供更高级别控制结构和错误检查的任务。

从20世纪60年代到70年代末期，现在使用的主要语言范式得到了发展：

APL，引入了阵列编程并影响了函数式编程。ALGOL，完善了结构化程序编程和语言规范学科；《ALGOL 60算法语言修订报告（Revised Report on the Algorithmic Language ALGOL 60）》成为后来编写语言规范的模型。Lisp，于1958年实现，是第一个动态类型的函数式编程语言。在20世纪60年代，Simula是第一种支持面向对象编程的语言；在20世纪70年代中期，Smalltalk采用了第一种“纯粹的”面向对象语言。C是在1969年和1973年之间开发的，作为Unix操作系统的系统编程语言，并且仍然很受欢迎。Prolog，设计于1972年，是第一个逻辑编程语言。1978年，ML语言在Lisp之上构建了一个多态类型系统，开创了静态类型的函数式编程语言。这些语言中的每一种都产生了后代，大多数现代编程语言至少在其祖先中至少有一种。

20世纪60年代和70年代也对结构化编程的优点进行了大量辩论，以及编程语言是否应该被设计为支持它。艾兹赫尔·戴克斯特拉（Edsger Dijkstra）在1968年出版的《ACM通讯》中发表的著名论文中指出，应该从所有更高级别的编程语言中删除GOTO语句。

### 整合和增长

20世纪80年代是相对稳定的年代。C++结合了面向对象和系统编程。美国政府对Ada语言进行了标准化，Ada是一种源自Pascal的系统编程语言，旨在供国防承包商使用。日本和其他地方花费了大量资金来研究所谓的“第五代”语言，这些语言包含了逻辑编程结构。所有这些研究与运动都没有发明新范式，而是阐述了前几十年发明的思想。

在20世纪80年代，用于编程大规模系统的语言设计的一个重要趋势是更多地关注模块或大规模组织代码单元的使用。Modula-2、Ada和ML都是在20世纪80年代开发的著名模块系统，它们通常与通用编程结构相结合。

20世纪90年代中期互联网的快速发展为新语言创造了机会。Perl语言最初是1987年首次发布的Unix脚本工具，在动态网站中很常见。Java开始用于服务器端编程，字节码虚拟机在商业环境中再次流行，承诺一次编写，随处运行（UCSD Pascal在20世纪80年代早期流行一段时间）。这些发展并不是根本新颖的，而是对许多现有语言和范例的改进（尽管它们的语法通常基于C系列编程语言）。

在工业和研究领域，编程语言的演变仍在继续。当前的方向包括安全性和可靠性验证，新型模块化和数据库集成，如Microsoft的LINQ。

第四代编程语言是计算机编程语言，旨在提供比第三代更高级别的内部计算机硬件细节抽象。第五代编程语言是基于使用给予程序的约束来解决问题的编程语言，而不是使用程序员编写的算法。

## 语言分类

### 机器语言

用机器语言编写程序，编程人员要首先熟记所用计算机的全部指令代码和代码的涵义。手编程序时，程序员得自己处理每条指令和每一数据的存储分配和输入输出，还得记住编程过程中每步所使用的工作单元处在何种状态。这是一件十分繁琐的工作，编写程序花费的时间往往是实际运行时间的几十倍或几百倍。而且，编出的程序全是由0和1的指令代码。直观性差，还容易出错。除了计算机生产厂家的专业人员外，绝大多数程序员已经不再去学习机器语言了。

### 汇编语言

为了克服机器语言难读、难编、难记和易出错的缺点，人们就用与代码指令实际含义相近的英文缩写词、字母和数字等符号来取代指令代码（如用 ADD 表示运算符号“+”的机器代码），于是就产生了汇编语言。所以说，汇编语言是一种用助记符表示的仍然面向机器的计算机语言。汇编语言亦称符号语言。汇编语言由于是采用了助记符号来编写程序，比用机器语言的二进制代码编程要方便些，在一定程度上简化了编程过程。汇编语言的特点是用符号代替了机器指令代码。而且助记符与指令代码一一对应，基本保留了机器语言的灵活性。使用汇编语言能面向机器并较好地发挥机器的特性，得到质量较高的程序。

汇编语言中由于使用了助记符号，用汇编语言编制的程序送入计算机，计算机不能象用机器语言编写的程序一样直接识别和执行，必须通过预先放入计算机的“汇编程序”的加工和翻译，才能变成能够被计算机识别和处理的二进制代码程序。用汇编语言等非机器语言书写好的符号程序称源程序，运行时汇编程序要将源程序翻译成目标程序。目标程序是机器语言程序，它一经被安置在内存的预定位置上，就能被计算机的 CPU 处理和执行。

汇编语言像机器指令一样，是硬件操作的控制信息，因而仍然是面向机器的语言，使用起来还是比较繁琐费时，通用性也差。汇编语言是低级语言。但是，汇编语言用来编制系统软件和过程控制软件，其目标程序占用内存空间少，运行速度快，有着高级语言不可替代的用途。

### 高级语言

不论是机器语言还是汇编语言都是面向硬件的具体操作的，语言对机器的过分依赖，要求使用者必须对硬件结构及其工作原理都十分熟悉，这对非计算机专业人员是难以做到的，对于计算机的推广应用是不利的。计算机事业的发展，促使人们去寻求一些与人类自然语言相接近且能为计算机所接受的语意确定、规则明确、自然直观和通用易学的计算机语言。这种与自然语言相近并为计算机所接受和执行的计算机语言称高级语言。高级语言是面向用户的语言。无论何种机型的计算机，只要配备上相应的高级语言的编译或解释程序，则用该高级语言编写的程序就可以通用。

如今被广泛使用的高级语言有 Java、Python、C、JavaScript 等。这些语言都是属于系统软件。

计算机并不能直接地接受和执行用高级语言编写的源程序，源程序在输入计算机时，通过“翻译程序”翻译成机器语言形式的目标程序，计算机才能识别和执行。这种“翻译”通常有两种方式，即编译方式和解释方式。编译方式是：事先编好一个称为编译程序的机器语言程序，作为系统软件存放在计算机内，当用户由高级语言编写的源程序输入计算机后，编译程序便把源程序整个地翻译成用机器语言表示的与之等价的目标程序，然后计算机再执行该目标程序，以完成源程序要处理的运算并取得结果。解释方式是：源程序进入计算机时，解释程序边扫描边解释作逐句输入逐句翻译，计算机一句句执行，并不产生目标程序。PASCAL、FORTRAN、COBOL 等高级语言执行编译方式；BASIC 语言则以执行解释方式为主；而 PASCAL、C 语言是能书写编译程序的高级程序设计语言。每一种高级（程序设计）语言，都有自己人为规定的专用符号、英文单词、语法规则和语句结构（书写格式）。高级语言与自然语言（英语）更接近，而与硬件功能相分离（彻底脱离了具体的指令系统），便于广大用户掌握和使用。高级语言的通用性强，兼容性好，便于移植。

# 人工智能概述

人工智能是计算机科学的一个分支，她企图了解智能的实质，并产生一种新的能以人类智能相似的方式做出反应的智能机器，该领域的研究包括机器人、语言识别、图像识别、自然语言处理和专家系统。

## 人工智能的发展

谈到人工智能，就不能不提到鼻祖式人物：图灵。1936年，英国数学家、逻辑学家阿兰·麦席森·图灵（1912~1954）提出了一种抽象的计算模型——图灵机(Turing Machine），用纸带式机器来模拟人们进行数学运算的过程，图灵本人被视为计算机科学之父。

1959年，图灵发表了一篇划时代的论文《计算机器与智能》，文中提出了人工智能领域著名的图灵测试——如果电脑能在5分钟内回答由人类测试者提出的一系列问题，且其超过30%的回答让测试者误认为是人类所答，则电脑就通过测试并可下结论为机器具有智能。

图灵测试的概念极大影响人工智能对于功能的定义，在这个途径上，卡内基·梅隆大学两位科学家纽厄尔（A.Newell）和西蒙（H.Simon）的“逻辑理论家”程序非常精妙地证明了罗素《数学原理》52道中的38道。西蒙宣称在10年之内，机器就可以达到和人类智能一样的高度。

第一批人工智能探索者找到共同的语言后，于整整60年前的1956年，在美国达特茅斯大学开了一次会，希望确立人工智能作为一门科学的任务和完整路径。与会者们也宣称，人工智能的特征都可以被精准描述，精准描述后就可以用机器来模拟和实现。后来普遍认为，达特茅斯会议标志着人工智能的正式诞生。

达特茅斯会议推动了全球第一次人工智能浪潮的出现，即为1956年到1974年。当时乐观的气氛弥漫着整个学界，在算法方面出现了很多世界级的发明，其中包括一种叫做增强学习的雏形（即贝尔曼公式）现在常听到的深度学习模型，其雏形叫做感知器，也是在那几年间发明的。除了算法和方法论有了新的进展，在第一次浪潮中，科学家们还造出了聪明的机器。

## 人工智能第一次浪潮和寒冬

在80年代出现了人工智能数学模型方面的重大发明，其中包括著名的多层神经网络（1986）和BP反向传播算法（1986）等，也出现了能与人类下象棋的高度智能机器（1989）。

但在1974至1980年人工智能进入冬天，因为人们发现逻辑证明器、感知器、增强学习等等只能做很简单、非常专门且很窄的任务，稍微超出范围就无法应对。这里面存在两方面局限：一方面，人工智能所基于的数学模型和数学手段被发现有一定缺陷；另一方面，有很多计算复杂度以指数程度增加，所以成为了不可能完成的计算任务。

## 现代PC“促成”第二次人工智能寒冬

然而，1987年到1993年现代PC的出现，让人工智能的寒冬再次降临。相比于现代PC，专家系统被认为古老陈旧而非常难以维护。于是，政府经费开始下降，寒冬又一次来临。

## 现代AI的发展

现代AI的曙光发生在这个阶段，出现了新的数学工具、新的理论和摩尔定律。人工智能也在确定自己的方向，其中一个选择就是要做实用性、功能性的人工智能，这导致了一个新的人工智能路径。由于对于人工智能任务的明确和简化，带来了新的繁荣。

在新的数学工具方面，原来已经存在于数学或者其他学科的文献中的数学模型，被重新发掘或者发明出来。当时比较显著几个成果包括最近获得图灵奖的图模型以及图优化、深度学习网络等，都是大约在15年前重新被提出来，重新开始研究。

在新的理论方面，由于数学模型对自然世界的简化，有着非常明确的数理逻辑，使得理论分析和证明成为可能，可以分析出到底需要多少数据量和计算量来以得期望的结果，这对开发相应的计算系统非常有帮助。

在更重要的一方面，摩尔定律让计算越来越强大，而强大计算机很少被用在人工智能早期研究中，因为早期的人工智能研究更多被定义为数学和算法研究。当更强大的计算能力被转移到人工智能研究后，显著提高了人工智能的研究效果。

由于这一系列的突破，人工智能又产生了一个新的繁荣期。

## 人工智能商业化浪潮

2016年IBM在全球范围内倾全力推出的认知商业，才是真正意义上的人工智能商业化第一波浪潮。在深蓝成功后，IBM研究院进而挑战人工智能的深度问答，这是人工智能的一个重要分支，具有极为广阔的应用空间。

## 人工智能的关键技术

### 人机交互

关于人机交互，它最重要的方面研究人和计算机之间的信息交换，主要包括人到计算机和计算机到人的两部分信息交换，是人工智能领域的重要外围技术。人机交互是与认知心理学、人机工程学、多媒体技术、虚拟现实技术等密切相关的综合学科。传统的人与计算机之间的信息交换主要依靠交互设备进行，主要包括键盘、鼠标、操纵杆、数据服装、眼动跟踪器、位置跟踪器、数据手套、压力笔等输入设备，以及打印机、绘图仪、显示器、头盔式显示器、音箱等输出设备。人机交互技术除了传统的基本交互和图形交互外，还包括语音交互、情感交互、体感交互及脑机交互等技术。

### 云计算与大数据

关于大数据和云计算的关系人们通常会有误解。而且也会把他们混起来说，分别做一句话解释就是：云计算就是硬件资源的虚拟化；大数据就是海量数据的高效处理。另外，如果做一个更形象的解释，云计算相当于我们的计算机和操作系统，将大量的硬件资源虚拟化之后再进行分配使用。

就目前而言，要想发展好大数据，就离不开云计算，我们在进行大数据的时候同样也是离不开云计算的，于是很多人觉得大数据与云计算都有一定的关系，那么大家知道不知道大数据的云计算有什么关系呢？我们在这篇文章中给大家带来这个问题的答案。

首先我们说一下大数据的定义吧，大数据就是需要新处理模式才能具有更强的决策力、洞察发现力和流程优化能力来适应海量、高增长率和多样化的信息资产。同样也是一种规模大到在获取、存储、管理、分析方面大大超出了传统数据库软件工具能力范围的数据集合，具有海量的数据规模、快速的数据流转、多样的数据类型和价值密度低四大特征。

那么大数据的技术有什么意义呢？大数据的意义并不是在于掌握庞大的数据信息，而在于对这些含有意义的数据进行专业化处理。换而言之，如果把大数据比作一种产业，那么这种产业实现盈利的关键，在于提高对数据的优化能力，通过优化实现数据的增值。

而大数据与云计算的关系在技术上的联系也是密不可分。大数据必然无法用一台计算机进行处理，必须采用分布式架构。它的特色在于对海量数据进行分布式数据挖掘。但它必须依托云计算的分布式处理、分布式数据库和云存储、虚拟化技术。随着云时代的来临，大数据也吸引了越来越多的关注。分析师团队认为，大数据通常用来形容一个公司创造的大量非结构化数据和半结构化数据，这些数据在下载到关系型数据库用于分析时会花费很多的财力和物力。大数据分析常和云计算联系到一起，因为实时的大型数据集分析需要框架来向数十、数百或甚至数千的电脑分配工作。并且，大数据需要特殊的技术，以有效地处理大量的容忍经过时间内的数据。适用于大数据的技术，包括大规模并行处理数据库、数据挖掘、分布式文件系统、分布式数据库、云计算平台、互联网和可扩展的存储系统。

## 知识图谱

知识图谱属于AI领域的是一个分支，很多人觉得它和CV（Computer Vision，计算机视觉），ASR（Automatic Speech Recognition，语音识别），以及NLP（Natural Language Processing，自然语言处理）一样都是特指的某一项技术，其实这么理解并不准确，它应该算是多种技术融合后的一种综合型技术。

知识图谱的历史最早要追溯到2012年，由Google公司提出主要用于提升搜索引擎的检索效率，但随着其发展其背后更深刻意义，远不仅是提高检索效率这么简单，而是整个搜索引擎结构的整体转型：将传统基于关键字的搜索模型转向基于语义的搜索升级。

如今针对知识图谱的技术方案已被国内外多家搜索引擎公司所采用，如：美国的微软必应，中国的百度、搜狗等，都在在短短的一年内纷纷宣布了各自的知识图谱产品，足以看出这革新对整个搜索引擎界的整体影响。

但现在这项技术的应用并不仅拘泥于搜索引擎领域范围，很多的数据分析软件，CRM系统（Customer Relationship Management System，客户关系管理系统）也开始采用基于知识图谱的模式去处理数据，从而去深入发现数据更大的价值。

知识图谱从字面上看，可以拆分为知识+图谱，这样我们就可以理解：将需要的知识数据（结构化或非结构化数据）以图谱的形式进行展示，这种简单的过程也是知识图谱的构建过程。

知识图谱为互联网上海量、异构、动态的大数据表达、组织、管理以及利用提供了一种更为有效的方式，使得网络的智能化水平更高，更加接近于人类的认知思维。

目前，知识图谱已在智能搜索、深度问答、社交网络以及一些垂直行业中有所应用，成为支撑这些应用发展的动力源泉。

## 机器学习

机器学习（Machine Learning）是一门专门研究计算机怎样才能模拟或实现人类的学习行为，以获取新的知识或技能，使之不断改善自身的性能的学科。机器学习虽然发展了几十年，但还是存在很多没有良好解决的问题，例如图像识别、语音识别、自然语言理解、天气预测、内容推荐等等。机器主要通过大量的训练数据进行训练，程序不断地进行自我学习和修正来训练出一个模型，而模型的本质就是一堆参数用成千上万的参数来描述业务特点，从而接近人类的智力。

机器学习是一门多领域交叉学科，涉及统计学、系统辨识、逼近理论、神经网络、优化理论、计算机科学、脑科学等诸多领域。通过研究计算机怎样模拟或实现人类的学习行为，以获取新的知识或技能。通过知识结构的不断完善与更新来提升机器自身的性能，这属于人工智能的核心领域。基于数据的机器学习是现代智能技术中的重要方法之一，研究从观测数据（样本）出发寻找规律，利用这些规律对未来数据或无法观测的数据进行预测。AlphaGo就这项技术一个很成功的体现。

根据学习模式将机器学习分类为监督学习、无监督学习和强化学习等。根据学习方法可以将机器学习分为传统机器学习和深度学习。

深度学习是机器学习的一个子集。深度学习的前身是人工神经网络（Artificial Neural Network, ANN），它的基本特点就是模仿人脑神经元传递和处理信息的模式。

**有监督学习：**输入的训练数据有特征、有标记，在学习中就是找到特征与标记之间的映射关系，通过标记不断纠正学习中的偏差，使预测率不断提高。这种训练数据有标记的学习称为有监督学习。

**无监督学习：**让计算机自己去学习怎样做一些事情，所有训练数据没有标记，只有特征。无监督学习有两种思路：第一种，训练时不为其指定明确分类但数据会呈现聚群的结构，彼此相似的类型会聚集在一起。计算机把这些没有标记的数据分成一个个组合，就是聚类；第二种，在成功时采用某种激励制度，即强化学习。

**半监督学习：**训练数据中有一部分有标记有一部分无标记，没有标记的数量远远大于有标记的数量（这也符合现实）。它的基本规律是：数据的分布必然不完全随机，通过结合有标记的局部特征，以及大量没标记的数据的整体分布，可以得到比较好的分类结果。

## 自然语言处理

自然语言处理（Natural Language Processing, NLP）就是用计算机来处理、理解以及运用人类语言(如中文、英文等)，它属于人工智能的一个分支，是计算机科学与语言学的交叉学科，又常被称为计算语言学。由于自然语言是人类区别于其他动物的根本标志。没有语言，人类的思维也就无从谈起，所以自然语言处理体现了人工智能的最高任务与境界，也就是说，只有当计算机具备了处理自然语言的能力时，机器才算实现了真正的智能。

从研究内容来看，自然语言处理包括语法分析、语义分析、篇章理解等。从应用角度来看，自然语言处理具有广泛的应用前景。特别是在信息时代，自然语言处理的应用包罗万象，例如：机器翻译、手写体和印刷体字符识别、语音识别及文语转换、信息检索、信息抽取与过滤、文本分类与聚类、舆情分析和观点挖掘等，它涉及与语言处理相关的数据挖掘、机器学习、知识获取、知识工程、人工智能研究和与语言计算相关的语言学研究等。

值得一提的是，自然语言处理的兴起与机器翻译这一具体任务有着密切联系。机器翻译指的是利用计算机自动地将一种自然语言翻译为另外一种自然语言。由于人工进行翻译需要训练有素的双语专家，翻译工作非常耗时耗力。更不用说需要翻译一些专业领域文献时，还需要翻译者了解该领域的基本知识。世界上有超过几千种语言，而仅联合国的工作语言就有六种之多。如果能够通过机器翻译准确地进行语言间的翻译，将大大提高人类沟通和了解的效率。

自然语言处理的困难可以罗列出来很多，不过关键在于消除歧义问题，如词法分析、句法分析、语义分析等过程中存在的歧义问题，简称为消歧。而正确的消歧需要大量的知识，包括语言学知识（如词法、句法、语义、上下文等）和世界知识（与语言无关）。这带来自然语言处理的两个主要困难。首先，语言中充满了大量的歧义，这主要体现在词法、句法及语义三个层次上。歧义的产生是由于自然语言所描述的对象——人类活动非常复杂，而语言的词汇和句法规则又是有限的，这就造成同一种语言形式可能具有多种含义。另外一个方面，消除歧义所需要的知识在获取、表达以及运用上存在困难。由于语言处理的复杂性，合适的语言处理方法和模型难以设计。

目前，人们主要通过两种思路来进行自然语言处理，一种是基于规则的理性主义，另外一种是基于统计的经验主义。理性主义方法认为，人类语言主要是由语言规则来产生和描述的，因此只要能够用适当的形式将人类语言规则表示出来，就能够理解人类语言，并实现语言之间的翻译等各种自然语言处理任务。而经验主义方法则认为，从语言数据中获取语言统计知识，有效建立语言的统计模型。因此只要能够有足够的用于统计的语言数据，就能够理解人类语言。然而，当面对现实世界充满模糊与不确定性时，这两种方法都面临着各自无法解决的问题。例如，人类语言虽然有一定的规则，但是在真实使用中往往伴随大量的噪音和不规范性。理性主义方法的一大弱点就是鲁棒性差，只要与规则稍有偏离便无法处理。而对于经验主义方法而言，又不能无限地获取语言数据进行统计学习，因此也不能够完美地理解人类语言。二十世纪八十年代以来的趋势就是，基于语言规则的理性主义方法不断受到质疑，大规模语言数据处理成为目前和未来一段时期内自然语言处理的主要研究目标。统计学习方法越来越受到重视，自然语言处理中越来越多地使用机器自动学习的方法来获取语言知识。迈进二十一世纪，我们已经进入了以互联网为主要标志的海量信息时代，这些海量信息大部分是以自然语言表示的。一方面，海量信息也为计算机学习人类语言提供了更多的“素材”，另一方面，这也为自然语言处理提供了更加宽广的应用舞台。例如，作为自然语言处理的重要应用，搜索引擎逐渐成为人们获取信息的重要工具，涌现出以百度、谷歌等为代表的搜索引擎巨头；机器翻译也从实验室走入寻常百姓家，谷歌、百度等公司都提供了基于海量网络数据的机器翻译和辅助翻译工具；基于自然语言处理的中文（输入法如搜狗、微软、谷歌等输入法）成为计算机用户的必备工具；带有语音识别的计算机和手机也正大行其道，协助用户更有效地工作学习。总之，随着互联网的普及和海量信息的涌现，自然语言处理正在人们的日常生活中扮演着越来越重要的作用。

## 计算机视觉

计算机视觉，英文Computer Vision，简称CV。计算机视觉是一门研究如何使机器“看”的科学，更进一步的说，就是指用摄影机和电脑代替人眼对目标进行识别、跟踪和测量等。

计算机视觉的主要任务就是通过对采集的图片或视频进行处理以获得相应场景的信息。计算机视觉任务的主要类型有以下几种：

## 物体检测

物体检测是视觉感知的第一步，也是计算机视觉的一个重要分支。物体检测的目标，就是用框去标出物体的位置，并给出物体的类别。

物体检测和图像分类不一样，检测侧重于物体的搜索，而且物体检测的目标必须要有固定的形状和轮廓。图像分类可以是任意的目标，这个目标可能是物体，也可能是一些属性或者场景。

### 物体识别（狭义）

计算机视觉的经典问题便是判定一组图像数据中是否包含某个特定的物体，图像特征或运动状态。这一问题通常可以通过机器自动解决，但是到目前为止，还没有某个单一的方法能够广泛的对各种情况进行判定：在任意环境中识别任意物体。

现有技术能够也只能很好地解决特定目标的识别，比如简单几何图形识别、人脸识别、印刷或手写文件识别，或者车辆识别。而且这些识别需要在特定的环境中，具有指定的光照，背景和目标姿态要求。

## 图像分类

一张图像中是否包含某种物体，对图像进行特征描述是物体分类的主要研究内容。一般说来，物体分类算法通过手工特征或者特征学习方法对整个图像进行全局描述，然后使用分类器判断是否存在某类物体。

图像分类问题就是给输入图像分配标签的任务，这是计算机视觉的核心问题之一。这个过程往往与机器学习和深度学习不可分割。

## 物体定位

如果说图像识别解决的是**what**，那么，物体定位解决的则是**where**的问题。利用计算视觉技术找到图像中某一目标物体在图像中的位置，即定位。

目标物体的定位对于计算机视觉在安防、自动驾驶等领域的应用有着至关重要的意义。

## 图像分割

在图像处理过程中，有时会需要对图像进行分割来提取有价值的用于后继处理的部分，例如筛选特征点，或者分割一或多幅图片中含有特定目标的部分等。

图像分割指的是将数字图像细分为多个图像子区域（像素的集合，也被称作超像素）的过程。图像分割的目的是简化或改变图像的表示形式，使得图像更容易理解和分析。更精确地说，图像分割是对图像中的每个像素加标签的一个过程，这一过程使得具有相同标签的像素具有某种共同视觉特性。

图像语意分割是一个像素级别的物体识别，即每个像素点都要判断它的类别。它和检测的区别是，物体检测是一个物体级别的，他只需要一个框，去框住物体的位置，而通常分割是比检测要更难的问题。

计算机视觉是通过创建人工模型来模拟本由人类执行的视觉任务。其本质是模拟人类的感知与观察的一个过程。这个过程不止识别，而是包含了一系列的过程，并且最终是可以在人工系统中被理解和实现的。

# 智能机器人

智能机器人之所以叫智能机器人，这是因为它有相当发达的“大脑”。在脑中起作用的是中央处理器，这种计算机跟操作它的人有直接的联系。最主要的是，这样的计算机可以进行按目的安排的动作。正因为这样，我们才说这种机器人才是真正的机器人，尽管它们的外表可能有所不同。

## 类脑智能

类脑智能又称为类脑计算，上世纪80年代末，美国科学家卡沃·米德（Carver Mead）首次提出类脑计算的概念。类脑计算这一想法摆脱了传统的计算模式，模仿人类神经系统的工作原理，渴求开发出快速、可靠、低耗的运算技术。类脑智能是人工智能的终极目标，但研究类脑智能不可能复制人的大脑。类脑智能希望通过研究人类大脑的工作机理并模拟出一个和人类一样具有思考、学习能力的机器人。类脑智能技术充分学习人脑的思维模式，从仿生角度努力寻求人工智能的突破。这一热门学科前景诱人，应用范围广阔。科学家们曾预言一个国家类脑智能的发展水平将极大程度影响该国在军事、工业等众多行业的发展，因此类脑智能技术的发展显得尤为重要与急迫。

## 人工智能的应用

人工智能应用（Applications of Artificial Intelligence）的范围很广，包括：医药，诊断，金融贸易，机器人控制，法律，科学发现和玩具。许多千种人工智能应用深入于每种工业的基础。90年代和21世纪初，人工智能技术变成大系统的元素；但很少人认为这属于人工智能领域的成就。

### 人工智能在金融领域的应用

银行用人工智能系统组织运作，金融投资和管理财产。2001年8月在模拟金融贸易竞赛中机器人战胜了人。

金融机构已长久用人工神经网络系统去发觉变化或规范外的要求，银行使用协助顾客服务系统；帮助核对帐目，发行信用卡和恢复密码等

### 人工智能在医疗领域的应用

医学临床可用人工智能系统组织病床计划，并提供医学信息。

人工神经网络用来做临床诊断决策支持系统。用人工智能在医学方面还有下列潜在可能：计算机帮助解析医学图像。这样系统帮助扫描数据图像，从计算X光断层图发现疾病，典型应用是发现肿块。

心脏声音分析。

### 人工智能在家居领域的应用

人工智能在智能家居场景中，一方面将进一步推动家居生活产品的智能化，包括照明系统、影音系统、能源管理系统、安防系统等，实现家居产品从感知到认知到决策的发展，另一方面在于智能家居系统的建立，搭载人工智能的多款产品都有望成为智能家居的核心，包括机器人、智能音箱、智能电视等产品，智能家居系统将逐步实现家居自我学习与控制，从而提供针对不同用户的个性化服务。

### 人工智能在汽车领域的应用

基于AI的自动驾驶，自动驾驶车辆所需的处理能力是巨大的，而传统的计算机根本无法胜任这项任务。因为自动驾驶不是遵循一套规则或简单的算法，它涉及到深度学习等，换句话说，就涉及人工智能。越来越多的汽车制造商和初创企业都在开发人工智能应用程序，目前在自动驾驶领域有两家公司处于领先地位：谷歌和特斯拉。

基于AI的云服务，人工智能和云服务也是形影不离，相互作用的。无论如何，汽车都需要通过大量的数据来分析完成相应的任务。人工智能云服务的应用确保了数据的有效性，更充分体现了大数据的潜在价值。

基于AI的汽车保险，保险公司利用人工智能实时进行风险评估。

基于AI的汽车制造，人工智能不仅改变了汽车的功能，也改变了汽车的制造方式。新出现的是与人类并肩作战的智能机器人。例如2018年初，起亚汽车开始与现代合作，为其装配线开发可穿戴工业机器人。背心外骨骼(H-VEX)和无椅外骨骼(H-CEX)可穿戴机器人帮助保护工作人员的膝盖、背部和颈部，同时赋予他们更大的灵活性与更强的力量

基于AI的驾驶员监控，以色列初创企业eyeSight借助先进的TOF摄像机和红外传感器，检测驾驶员的行为，

## 人工智能在零售领域的应用

人工智能在商业界中已经掀起了一阵风潮，其系统的庞大市场规模越来越受到重视。具体体现为店面优化，供应链优化，营销策略的变化，手势识别等方面

## 人工智能在教育领域的应用

个性化学习，因材施教：分析内容，构建知识图谱，构建和优化内容模型，建立知识图谱，让用户可以更容易地、更准确地发现适合自己的内容。国外这方面的典型应用是分级阅读平台，推荐给用户适宜的阅读材料，并将阅读与教学联系在一起，文后带有小测验，并生成相关阅读数据报告，老师得以随时掌握学生阅读情况。

自动化辅导与答疑：AI除了应用于个性化学习方案的制定外，还落地在自动化辅导和答疑领域，这也成为了教师面授外的补充。

智能测评：随着信息化建设、人工智能的发展，大数据、文字识别、语音识别、语义识别，使得规模化的自动批改和个性化反馈走向现实。如何利用人工智能减轻批改压力，实现规模化又个性化的作业反馈，是未来教育的重要攻克点

模拟和游戏化教学平台：寓教于乐也是现代教育理念之一。

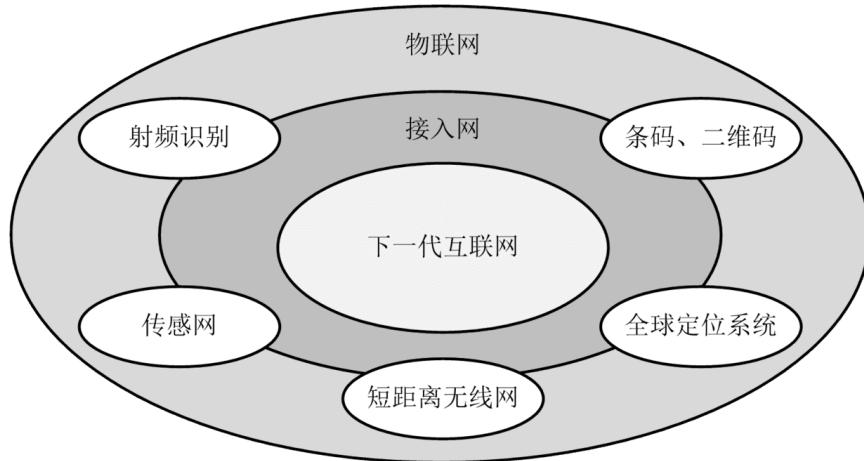
教育决策：AI能做的远远不止大学专业选择的分析决策，AI帮助决策将越来越多地影响我们生活的方方面面

幼儿早教机器人：早教的未来在移动和智能。儿童机器人的门槛不在技术这块，而在于内容、交互方式。

## 物联网概述

### 物联网的发展

物联网的概念英文术语为**Internet of Things**。物联网是指按照约定的协议，将具有感知、通信、计算功能的智能物体、系统、信息资源互联起来，实现对物理世界广泛感知、可靠传输、智慧处理的智能服务系统。20世纪90年代有关物联网的研究开始萌芽，此后其概念不断地演进和发展。1999年美国麻省理工学院的Kevin Ashton和他的同事首次提出**Internet of Things**的概念。同年，中科院启动了传感网的研究，并取得了一些科研成果，建立了一些适用的传感网。2005年，国际电信联盟（ITU）在《The Internet of Things》报告中对物联网概念进行扩展。无所不在的物联网通信时代即将来临。



进入2009年，物联网得到了真正的起步。

**2009年1月**，奥巴马就任美国总统后，与工商界代表举行了一次圆桌会议。IBM公司代表提出了智慧地球的研究计划，建议新政府投资新一代的智慧型基础设施。当年，美国将新能源和物联网列为振兴经济的两大核心武器；

**2009年8月**，温家宝总理在视察无锡时提出建设“感知中国”中心，物联网被正式列为国家五大新兴战略性产业之一，写入“政府工作报告”，物联网在中国受到了全社会极大的关注；

**2009年9月**，欧盟第七框架下RFID和物联网研究项目簇（European Research Cluster on the Internet of Things）发布了《物联网战略研究路线图》研究报告，定义为基于标准的和可互操作的通信协议且具有自配置能力的动态的全球网络基础架构。至此，物联网真正的建立了起来，随后的几年物联网开始迈上了新的台阶。我们要学习物联网，必然要从技术层面入手，当下从技术的角度来理解，物联网的发展共分为4个阶段，具体如下图所示：



物联网当今在中国的受关注程度是在美国、欧盟及其他各国不可比拟的，中国在物联网理念和应用方面可以说是已经走在了世界的前面。近年来，电子信息技术和人工智能应用的发展促使物联网的内涵和外延有了很大的拓展，物联网已经表现为信息技术和通信技术的发展融合，是信息社会发展的趋势。

## 物联网的关键技术

物联网的关键技术主要包括三个方面：各类终端实现“全面感知”；电信网、互联网等融合实现“可靠创术”；云计算等技术对海量数据的“智能处理”。通过这三个方面的有机结合实现各类资源的“虚拟”和“共享”。

### 全面感知

全面感知是指利用射频识别（RFID）、传感器、定位器和二维码等手段随时随地对物体进行信息采集和获取。



### 可靠传输

可靠传输是指通过各种电信网络与互联网的融合，对接收到的感知信息进行实时远程传送，实现信息的交互和共享，并进行各种有效的处理。在这一过程中，通常需要用到现有的电信运行网络，包括无线和有线网络。由于传感器网络是一个局部的无线网，因此无线移动通信网、4G、5G网络是作为承载物联网的一个有力的支撑。



## 智能处理

物联网是一个智能的网络，面对采集的海量数据，必须通过智能分析和处理才能实现智能化。智能处理是指利用云计算、数据挖掘、模糊识别等各种智能计算技术，对随时接收到的跨地域、跨行业、跨部门的海量数据和信息进行分析处理，提升对物理世界、经济社会各种活动和变化的洞察力，实现智能化的决策和控制。



## 边缘计算

边缘计算起源于传媒领域，是指在靠近物或数据源头的一侧，采用网络、计算、存储、应用核心能力为一体的开放平台，就近提供最近端服务。其应用程序在边缘侧发起，产生更快的网络服务响应，满足行业在实时业务、应用智能、安全与隐私保护等方面的基本需求。边缘计算处于物理实体和工业连接之间，或处于物理实体的顶端。而云端计算，仍然可以访问边缘计算的历史数据。对物联网而言，边缘计算技术取得突破，意味着许多控制将通过本地设备实现而无需交由云端，处理过程将在本地边缘计算层完成。这无疑将大大提升处理效率，减轻云端的负荷。由于更加靠近用户，还可为用户提供更快的响应，将需求在边缘端解决。边缘计算五大优势应对物联网大数据挑战

## 实时计算，减少反应延迟

边缘计算分布式以及靠近设备端的特性注定它实时处理的优势，所以它能够更好的支撑本地业务实时处理与执行。

## 可靠性高，离线正常运作

家里的事情就不麻烦远在天边的云计算了，碰到没有网的情况下也一样能运行正常，边缘计算直接对终端设备的数据进行过滤和分析，节能省时效率还高，不受网络限制。

## 安全合规，满足隐私要求

制造、能源、公共事业等行业要实现智能化，需要整合机械、电子、ICT等跨行业技术，边缘计算首先要实现OT(Operation Technology)和IT领域的深度协作，并将行业专有技术与知识与ICT(Information and Communication Technology) 数字化技术相结合，满足多种用户需求，无论是什么样子的安全协议，他都将支持。

## 高性价比，节省存储运输成本

按照IDC（国际数据公司）的统计数据，到2020年将有超过500亿的终端与设备联网，未来超过50%的数据需要在网络边缘侧分析、处理与储存。边缘计算所面对的市场规模非常巨大，可以存储可运输，不用大型的服务器机房，照样能够运行。

## 灵活部署，新旧设备互通

不用担心新设备的诞生造成老设备相关接口的不兼容问题，无论你是智慧城市、智能家居、智慧医院、在线直播，到智能泊车、自动驾驶、无人机、智能制造还是其他，我们都可以灵活部署到每一个应用领域，且保证新旧设备互通互联。



如果说云计算就像是天上的云，看得见摸不着，像章鱼的大脑，边缘计算就类似于八爪鱼的那些小爪子，一个爪子就是一个小型的机房，靠近具体的实物。把云计算看作是大脑，那么边缘计算就像是大脑输出的神经触角，这些触角连接到各个终端运行各种动作。



阿里云边缘计算产品Link Edge已经问世。据说通过这款产品，开发者能够轻松将阿里云的边缘计算能力部署在各种智能设备和计算节点上，比如车载中控、工业流水线控制台、路由器等。另外基于生物识别技术的智能云锁利用本地家庭网关的计算能力，可实现无延时体验，断网了还能开锁，避免“被关在自己家门外”的尴尬。云与边缘的协同计算，还能实现场景化联动，一推开门，客厅的灯就自动打开迎接你回家。



## 传感器

在物联网中，传感器主要负责接收物品讲话的内容。传感器技术是从自然信源获取信息并对获取的信息进行处理、变换、识别的一门多学科交叉的现代科学与工程技术，它涉及传感器、信息处理和识别的规划设计、开发、制造、测试、应用及评价改进活动等内容。计算机类似于人的大脑，但仅有大脑而没有感知外界信息的“五官”显然是不够的，计算机也需要它们的五官——传感器。

传感器也就是我们物联网系统中主要用于采集物理世界中发生的物理事件和数据，包括各类物理量、标识、音频和视频数据等。物联网数据采集涉及的技术有多种，主要包括传感器、RFID、多媒体信息采集、实时定位等。传感器网络组网和协同信息处理技术实现传感器、RFID等数据采集技术所获取数据的短距离传输、自组织组网及多个传感器对数据的协同信息处理过程。物联网传感器解决的就是人类世界和物理世界的数据获取问题，包括以下几点：



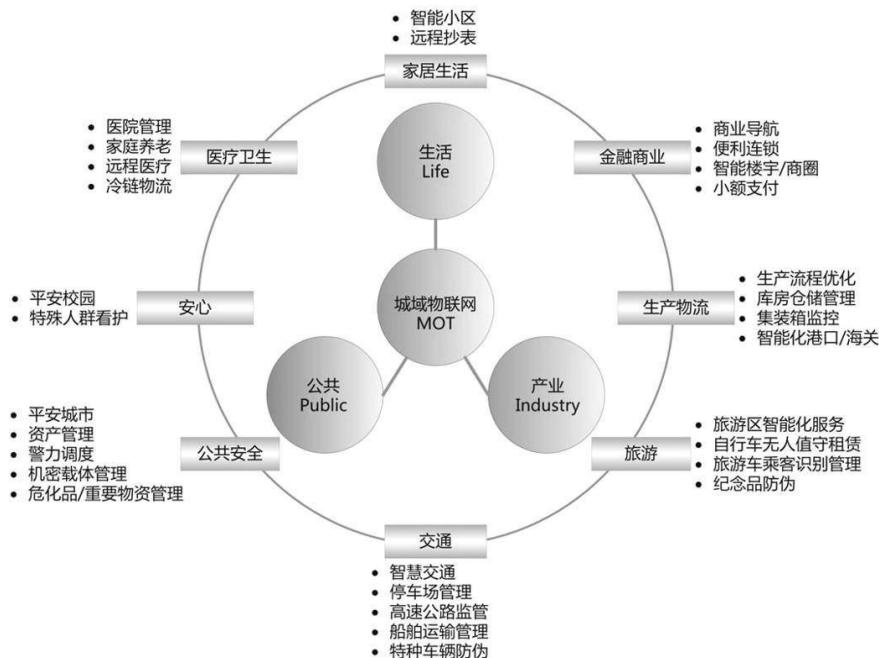
传感器是一种检测装置，能感受到被检测的信息，并能将检测感受到的信息，按一定规律变换成为电信号或其他所需形式的信息输出，以满足信息的传输、处理、存储、显示、记录和控制等要求，如下图所示，它是实现自动检测和自动控制的首要环节，我们常用的包括以下几种：



## 物联网的应用

物联网作为一种新兴的信息技术正在迅速向各个行业蔓延，从家庭、医疗保健、物流、汽车、零售到工业制造，物联网产品技术将无处不在。例如，时下流行的共享单车，只要拿出手机扫一扫便可打开智能锁骑行，这些智能锁使用的就是物联网技术。

物联网是一个极其庞大的网络，它包罗万象，涉及各行各业，其实质就是让万事万物通过网络连接起来，实现众多超级智能化的应用。业内预测，未来20年内物联网设备接入数量将越过1万亿台，这意味着物联网产业蕴含着巨大的市场机遇。



目前物联网的应用主要包括：智能工业、智能农业、智能电网、智能家居、智慧校园、智慧医疗、智能交通、智能物流、环境监测等方面。从市场应用来看，智能工业占据物联网市场主要份额最大。物联网技术的应用提高了生产线过程检测、实时参数采集、生产设备监控、材料消耗监测的能力和水平。生产过程的智能监控、智能控制、智能诊断、智能决策、智能维护水平不断提高。钢铁企业应用各种传感器和通信网络，在生产过程中实现对加工产品的宽度、厚度、温度的实时监控，从而提高了产品质量，优化了生产流程。



## 物联网在家居领域的应用

智能家居是一个居住环境，是以住宅为平台安装有智能家居系统的居住环境，实施智能家居系统的过程称为智能家居集成。智能家居通过物联网技术将家中的各种设备连接到一起，提供家电控制、照明控制、窗帘控制、电话远程控制、室内外遥控、防盗报警、环境监测、暖通控制、红外转发以及可编程定时控制等多种功能和手段，让用户有更方便的手段来管理家庭设备。



通过触摸屏、无线遥控器、电话、互联网或者语音识别控制家用设备，更可以执行场景操作，使多个设备形成联动；另一方面，智能家居内的各种设备相互间可以通信，不需要用户指挥也能根据不同的状态互动运行，从而给用户带来最大程度的高效、便利、舒适与安全。与普通家居相比，智能家居不仅提供舒适宜人且高品位的家庭生活空间，还将传统家居环境中那些各自单独存在的设备联为一个整体，形成集系统、结构、服务、管理为一体的高效、安全、便利、环保的居住环境，提供全方位的信息交互功能，帮助家庭与外部保持信息交流畅通，优化人们的生活方式，帮助人们有效安排时间，增强家居生活的安全性，甚至为各种能源费用节约资金。



## 物联网在教育领域的应用

物联网再教育中的应用，目前来说分为具体的五个方面，分别是教学质量监控、学生健康状况监测、信息化教学应用、智慧管理、智慧校园等。

### 教学质量监控

通常是将物联网与现有教学平台集成，开发阅读器接口中间件，对于需要督导的自律性较差的学生，定时佩戴传感器手表、眼镜等记录学生的多重数据，如脑电图、血压、体温等生理信息及眼动、手部轻微移动等运动信息，引入心理学相关测试技术，得出学生的紧张程度、注意力状况、动脑情况等。将传感器获取的实时数据导入现有教学平台，老师根据这些反馈信息对学生进行有效的督促辅导。

### 学生健康状况监测

通过门式晨检机感知学生的健康信息，自动采集体温指标，当学生体温异常时，可通过短信等通知家长与老师，当学校出现一定数量体温异常案例时，即可通过应急联动机制，将信息传至医疗机构跟踪处理，防止出现集体疫情；而通过为学生配置运动传感器，可以系统感知其运动指标，避免学校只培养“书呆子”。

### 信息化教学应用

利用物联网建立泛在学习环境。可以利用智能标签识别需要学习的对象，并且根据学生的学习行为记录，调整学习内容。这是对传统课堂和虚拟实验的拓展，在空间上和交互环节上，通过实地考察和实践，增强学生的体验。例如生物课的实践性教学中需要学生识别校园内的各种植物，可以为每类植物粘贴带有二维码的标签，学生在室外寻找到这些植物后，除了可以知道植物的名字，还可以用手机识别二维码从教学平台上获得相关植物的扩展内容。

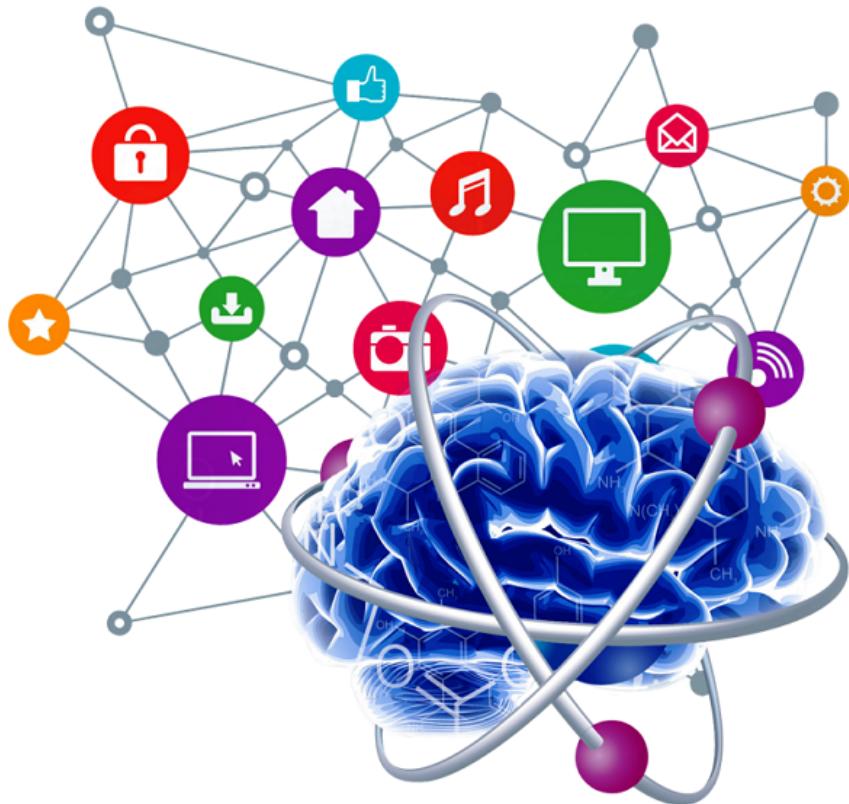
### 智慧管理

物联网在教育管理中可以用于人员考勤、图书管理、设备管理等方面。例如带有RFID标签的学生证可以监控学生进出各个教学设施的情况，以及行动路线。又如将RFID用于图书管理，可通过RFID标签可方便地找到图书，并且可以在借阅图书

的时候方便地获取图书信息而不用把书一本一本拿出来扫描。将物联网技术用于实验设备管理可以方便地跟踪设备的位置和使用状态，方便管理。

### 智慧校园

智能化教学环境，控制物联网在校园内还可用于校内交通管理、车辆管理、校园安全、智能建筑、学生生活服务等领域。例如，在教室里安装光线传感器和控制器，根据光线强度和学生的位置，调整教室内的光照度。控制器也可以和投影仪和窗帘导轨等设备整合，根据投影工作状态决定是否关上窗帘，降低灯光亮度。



## 开源硬件概述

开源硬件指与自由及开放原始码软件相同方式设计的计算机和电子硬件。开源硬件开始考虑对软件以外的领域开源，是开源文化的一部分。



### 开源硬件的发展

开源硬件（Open Source Hardware），是指与自由及开放源代码相同方式设计的计算机和电子硬件，是开源文化的一部分。开源文化源于20世纪70年代的黑客亚文化。到了90年代，随着Linux受到大众认可、Netscape浏览器开放源代码等一系列

科技事件，开源运动逐渐进入人们的视野。开源运动最早只有开源软件，并基于互联网进行传播。目前，我们日常生活中使用的手机操作系统安卓（Android）、电脑浏览器 Chrome都是属于开源软件。可以说，在我们的日常生活中，开源软件几乎无处不在。

开源软件推崇任何人都可以自由使用、复制、研究和改动的思想，深刻影响着开源文化的发展。开源硬件也在这种思想下应运而生。1997年，开放源代码促进会（OSI）推出开源硬件认证计划。1998年，大卫·弗里曼（David Freeman）提出开源硬件规范项目。1999年，非营利组织开放设计基金会（ODF）成立，一场开源硬件的运动悄然发生。

开源运动的一个核心是用户可以自行制造产品，无须支付任何费用。它的长足发展成为创客运动兴起的一个重要的技术因素，被誉为创客之父的克里斯安德森（Chris Anderson）在其著作《创客：新工业革命》中，将在开源社区中分享设计成果、开展合作的文化规范与使用数字桌面工具设计新产品和通过设计传给商业制造服务商或自行制造称为创客运动的3个变革性共同点。

## 流行的开源硬件

### Jetson Nano

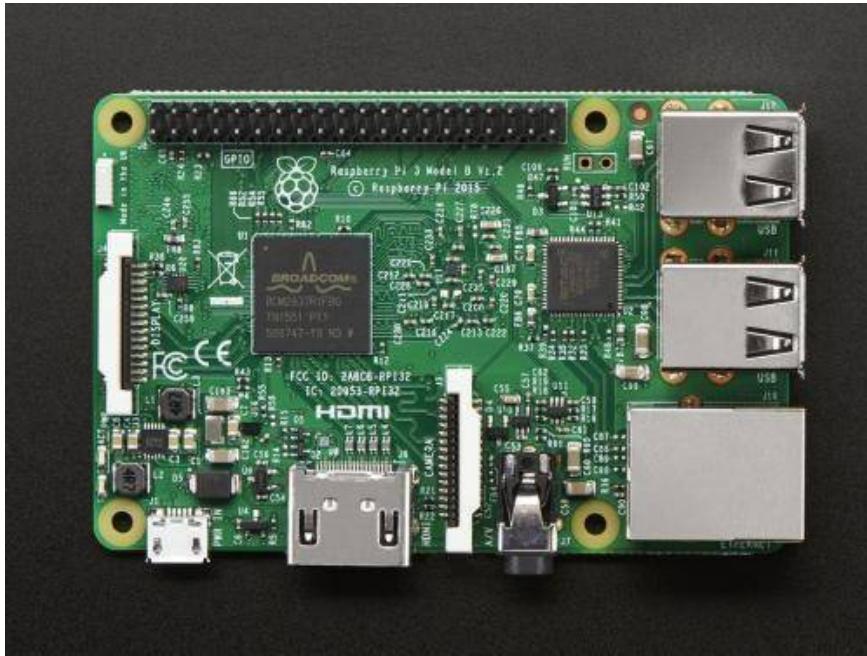
Jetson Nano是英伟达（NVIDIA）公司提供的一款面向AI的高性能低功率的开发板。具有HDMI、DP、以太网口、USB3.0、GPIO等多个接口。它开启了嵌入式物联网应用程序的新领域。Jetson Nano支持高分辨率传感器，可以并行处理多个传感器，并且可以在每个传感器流上运行多个现代神经网络。它内置了英伟达的硬件加速系统和OpenCV，还支持许多流行的人工智能框架，使开发人员可以很容易地将他们喜欢的模型和框架集成到产品中。Jetson Nano使所有人都能更容易地访问人工智能，并由相同的基础架构和软件提供支持。



### 树莓派（Raspberry Pi）

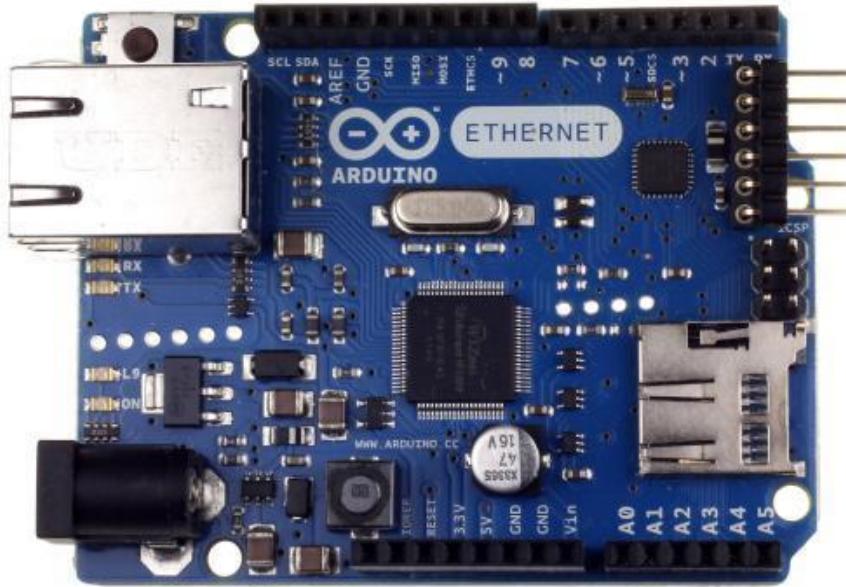
树莓派是一款针对电脑业余爱好者、教师、学生以及小型企业等用户的迷你电脑。树莓派基于Linux系统，并采用ARM架构处理器作为主芯片，也提供了USB与以太网接口。树莓派没有板载存储芯片，仅留有SD卡座，因而运行树莓派需要提供SD卡。树莓派尤其适合于需要支持用户界面的场合，因为它拥有HDMI输出。HDMI接

口意味着我们可以将树莓派直接接入到电视或其他显示屏上，从而以低成本构建web浏览设备来支持与用户的交互。换句话说，树莓派可以看成一台功能相对完备的电脑，尽管性能不高。



## Arduino

Arduino是一款便捷灵活、方便上手的开源电子原型台，包含硬件（各种型号的Arduino板）和软件（Arduino IDE）两部分，由一个欧洲开发团队于2005年冬季开发。Arduino使用Atmel公司的一款微处理器作为主芯片，具有体积小、价格实惠等特性。不仅如此，Arduino在设计之初就考虑到了与不同的外设进行交互，在与现有的电子元件例如传感器或者其他控制器件、LED、步进马达等连接时，几乎不需要增加支持电路。当然，Arduino也可以独立运行，并与软件进行交互。同时，Arduino IDE基于processing IDE开发，灵活且简单。开发语言Arduino语言基于wiring语言开发，是对avr-gcc库的二次封装，不要求开发者有太多的编程基础，可以说Arduino对初学者非常友好。



## ESPRESSIF

ESP8266EX是一个价格低廉的开发板，包含WiFi模块和GPIO，可以连接传感器、舵机、马达等各种设备。使用Arduino IDE进行开发编程。可通过网络、串口和蓝牙等多种方式进行通信。该芯片可工作于三种模式下，分别是：AP模式，station模式以及混合模式，通过常用的AT指令进行控制。其具有以下四个主要特性：

### 性能稳定

ESP8266EX的工作温度范围大，且能够保持稳定的性能，能适应各种操作环境。

### 高度集成

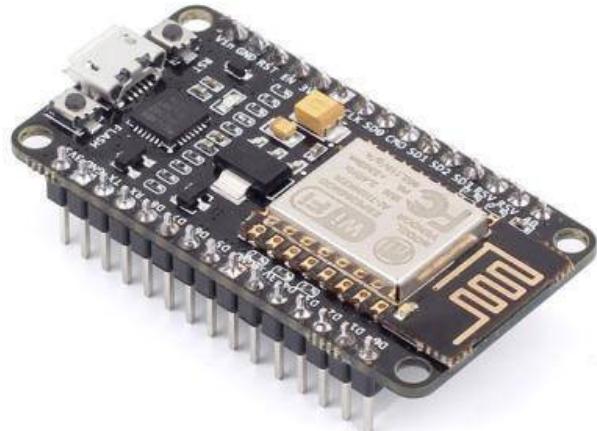
ESP8266EX集成了32位Tensilica处理器、标准数字外设接口、天线开关、射频、功率放大器、低噪放大器、过滤器和电源管理模块等，仅需很少的外围电路，可将所占PCB空间降低。

### 低功耗

ESP8266EX专为移动设备、可穿戴电子产品和物联网应用而设计，通过多项专有技术实现了超低功耗。ESP8266EX具有的省电模式适用于各种低功耗应用场景。

### 32位Tensilica处理器

ESP8266EX内置超低功耗Tensilica L106 32位RISC处理器，CPU时钟速度最高可达160MHz，支持实时操作系统（RTOS）和WiFi协议栈，可将高达80%的处理能力留给应用编程和开发。



© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-03-13

# 环境准备

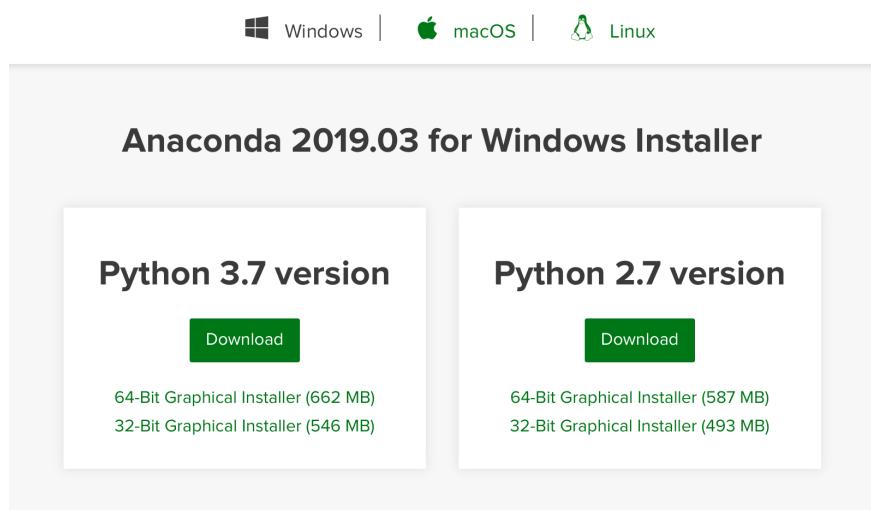
相关软件及资源，请[点击这里](#)来下载  
所需的用户名和密码均为 sli  
也可以选择在下方提供的官方网址下载  
推荐按顺序依次安装以下软件，以避免因依赖问题报错

## 在笔记本电脑上的操作

### Anaconda

Anaconda是一个Python环境管理软件。在Windows, Mac、Linux上均可以方便安装

下载链接：<https://www.anaconda.com/distribution>



选择适合自己的操作系统，并选择Python 3.7版本

### 基本命令概览

#### 创建环境

```
conda create -n 环境名字
//例如:
conda create -n py27 python=2.7
//表示创建一个名字为py27，运行python2.7的虚拟环境
//后面的python=2.7是可选输入
//不输入时默认环境是python3
```

#### 进入环境

```
conda activate 环境名字  
//例如:  
conda activate py27
```

## 安装指定包

```
conda install 包名  
//例如: 安装OpenCV  
conda install opencv
```

## 其他命令

```
//退出环境  
conda deactivate  
//列出环境  
conda-env list  
//删除环境  
conda-env remove -n 环境名字
```

## setup

1. 下载安装Anaconda
2. macOS用户打开 终端 , Windows用户在开始菜单打开 Anaconda Prompt
3. 创建并进入环境 (python版本为默认的3.x)
4. 在新环境中安装TensorFlow和OpenCV (若电脑有独立显卡应安装GPU版本的TensorFlow)

```
//创建一个名字为learn-ai的虚拟环境  
conda create -n learn-ai  
  
//激活learn-ai虚拟环境  
conda activate learn-ai  
  
//无独立显卡的电脑使用这条命令  
conda install tensorflow  
//有独立显卡的电脑使用这条命令  
conda install tensorflow-gpu  
  
//安装opencv  
conda install opencv  
  
//安装git命令  
conda install git
```

## VSCode

VSCode是微软出品的免费代码编辑软件。在Windows、Mac、Linux上均可以方便安装

下载链接: <https://code.visualstudio.com>

## setup

1. 下载安装VSCode
2. 安装插件 Settings Sync



3. 输入 `shift + Alt + D`, 输入 GitHub Token 和 Gist Token([点击获取](#)), 即可从服务端同步设置。免去自己配置的麻烦

## CP2102驱动

这个驱动用于使用USB串口连接esp8266

注意选择对应的操作系统和版本进行下载和安装

下载链接: <https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

## Arduino IDE

Arduino IDE (Integrated Development Environment, 集成开发环境) 是针对 Arduino 控制板的编程和下载平台。在 Windows, Mac、Linux 上均可以方便安装, Arduino 项目文件的后缀是 `*.ino`

项目文件应在与项目名相同的文件夹中

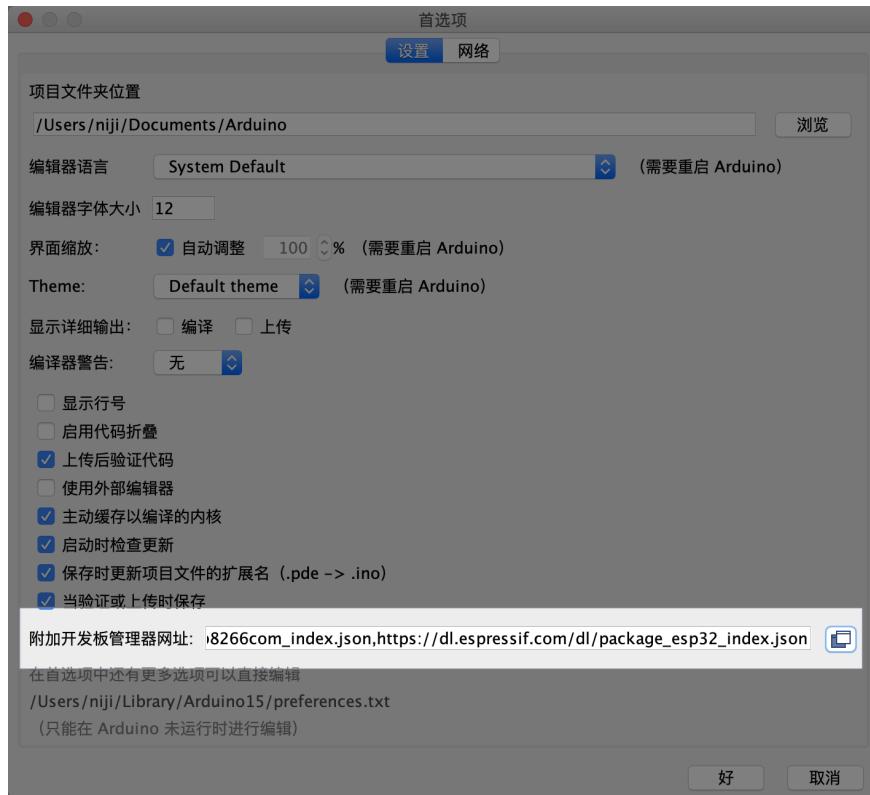
下载链接: <https://www.arduino.cc/en/Main/Software>

## setup

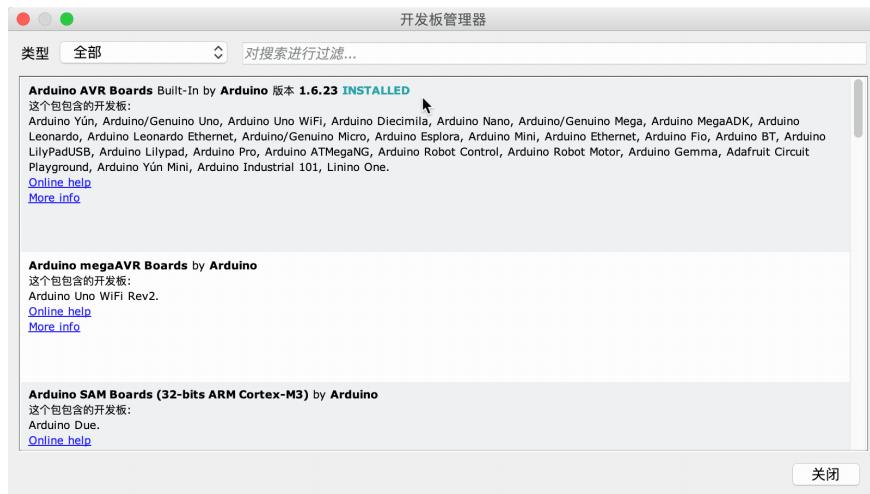
1. 下载安装 Arduino IDE

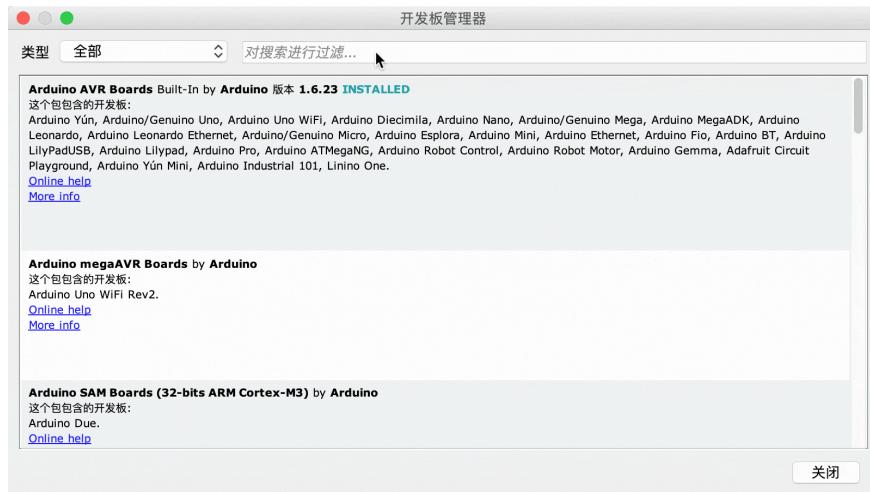
2. 在 `文件 -- 首选项 -- 附加开发板管理器网址` 一栏中输入

[https://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](https://arduino.esp8266.com/stable/package_esp8266com_index.json), [https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json) 重启 IDE

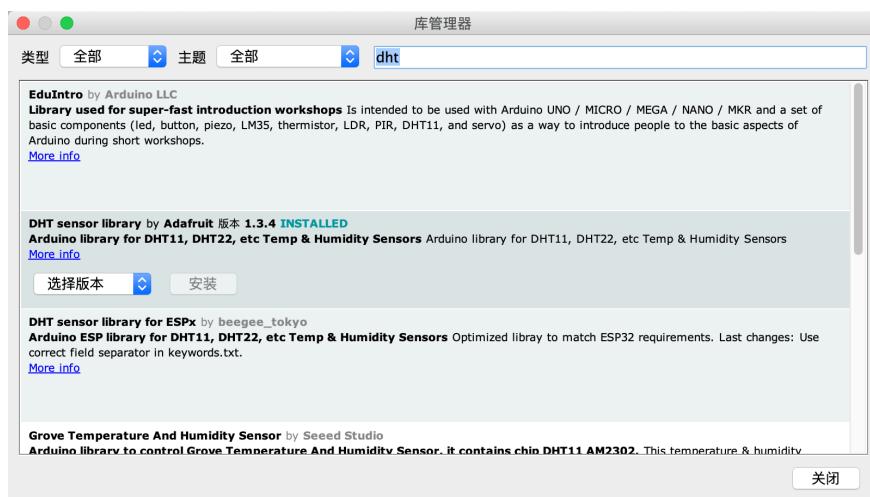


3. 在 工具 -- 开发板 -- 开发板管理器 中分别搜索esp8266和esp32,点击对应项进行安装





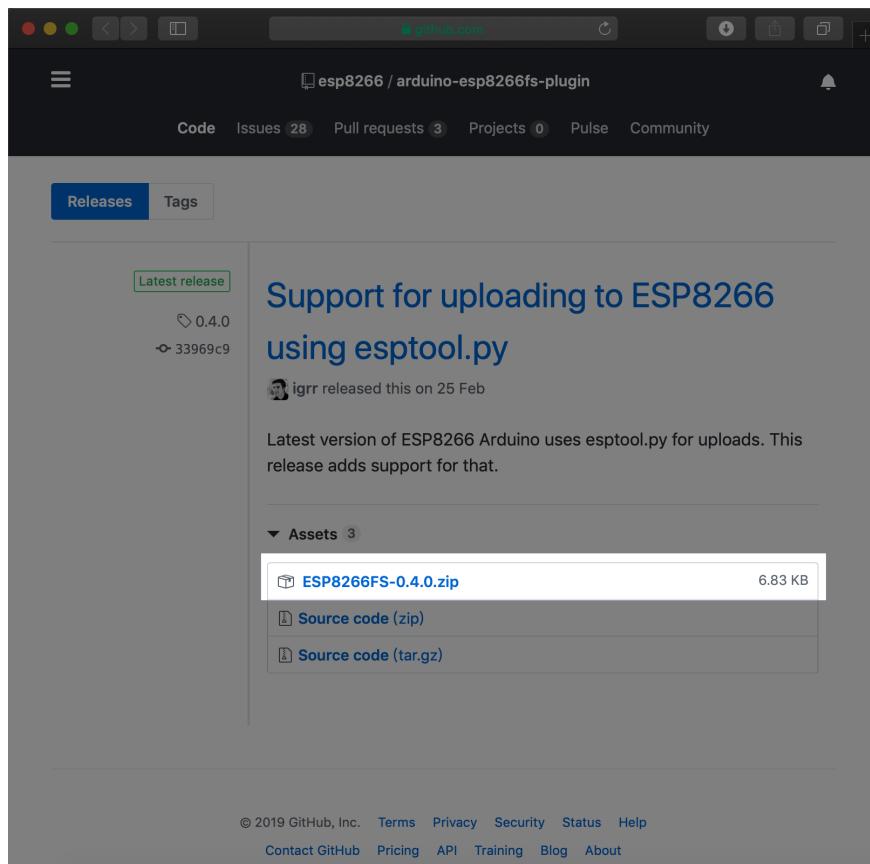
4. 在 工具 -- 管理库 中搜索DHT,选择DHT sensor library by Adafruit



5. 在 工具 -- 管理库 中搜索adafruit,选择Adafruit Unified Sensor by Adafruit



6. 打开链接<https://github.com/esp8266/arduino-esp8266fs-plugin/releases/tag/0.4.0>,选择.zip文件下载, 将解压后的文件夹复制到 Arduino安装目录/tools 文件夹, 然后重启IDE



默认的路径应该是这样: C:\Program Files

(x86)\Arduino\tools\esp8266FS\tool\esp8266fs.jar 如果安装成功，会在 工具 菜单下看到下图选项:

自动格式化	⌘T
项目存档	
修正编码并重新加载	
管理库...	⇧⌘I
串口监视器	⇧⌘M
串口绘图器	⇧⌘L
WiFi101 / WiFiNINA Firmware Updater	
<b>ESP8266 Sketch Data Upload</b>	
开发板: "NodeMCU 1.0 (ESP-12E Module)"	▶
Upload Speed: "921600"	▶
CPU Frequency: "80 MHz"	▶
Flash Size: "4M (3M SPIFFS)"	▶
Debug port: "Disabled"	▶
Debug Level: "无"	▶
lwIP Variant: "v2 Lower Memory"	▶
VTables: "Flash"	▶
Exceptions: "Disabled"	▶
Erase Flash: "Only Sketch"	▶
端口	▶
取得开发板信息	
编程器: "AVRISP mkII"	▶
烧录引导程序	

7. 设置开发板和端口



## 在Jetson Nano和树莓派上的操作

- 1.通电，连接显示器和键盘鼠标
- 2.连接WiFi
- 3.[下载课程文件](#)

## 下载课程文件

### Windows

- 打开 Anaconda Prompt
- 执行 `git clone https://github.com/nijisakai/learn-ai.git C:/learn-ai`
- 文件将被下载到C盘根目录下面的learn-ai文件夹

## macOS、Linux

- 打开 终端
- 执行 `git clone https://github.com/nijisakai/learn-ai.git ~/Desktop/learn-ai/`
- 文件将被下载到桌面下面的learn-ai文件夹

## 路由及WiFi设置

路由器设置SSID名字为**AI**, 密码为**raspberry**

路由器管理地址设置为<http://192.168.123.1>, 登录账号: **admin**, 登录密码:  
**admin**

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间: 2019-12-15

# 第1章 基础知识

## 本章主题：掌握基础

学习者通过学习本章内容，将熟悉掌握一些必备的相关技能。如Python基础，Linux常见命令操作，开源硬件的基础操作等。为后续的项目学习打好基础。后面的三、四、五、六章是四个综合项目，其对应的基础知识如图。



## 本章重点

了解Python基本语法规则、熟悉Linux基本操作，知道Web相关概念，会做简单的Web服务器，了解开源硬件的基本操作。

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook 修订时间：2020-03-13

# 第1节 Python语言基础

---

## Python简介

Python是一个高层次的结合了解释性、编译性、互动性和面向对象的脚本语言。

Python的设计具有很强的可读性，相比其他语言经常使用英文关键字，其他语言的一些标点符号，它具有比其他语言更有特色语法结构。

Python是一种解释型语言：这意味着开发过程中没有了编译这个环节。类似于 PHP和Perl语言。

Python是交互式语言：这意味着，您可以在一个Python提示符 >>> 后直接执行代码。

Python是面向对象语言：这意味着Python支持面向对象的风格或代码封装在对象的编程技术。

Python是初学者的语言：Python对初级程序员而言，是一种伟大的语言，它支持广泛的应用程序开发，从简单的文字处理到浏览器再到游戏。

## Python发展历史

Python是由Guido van Rossum在八十年代末和九十年代初，在荷兰国家数学和计算机科学研究所设计出来的。

Python本身也是由诸多其他语言发展而来的,这包括ABC、Modula-3、C、C++、Algol-68、SmallTalk、Unix shell和其他的脚本语言等等。

像Perl语言一样，Python源代码同样遵循GPL(GNU General Public License)协议。

现在Python是由一个核心开发团队在维护，Guido van Rossum仍然占据着至关重要的作用，指导其进展。

Python 2.7被确定为最后一个Python 2.x版本，它除了支持 Python 2.x语法外，还支持部分Python 3.1语法。

## Python特点

1.易于学习：Python有相对较少的关键字，结构简单，和一个明确定义的语法，学习起来更加简单。

2.易于阅读：Python代码定义的更清晰。

3.易于维护：Python的成功在于它的源代码是相当容易维护的。

4.一个广泛的标准库：Python的最大的优势之一是丰富的库，跨平台的，在UNIX，Windows和macOS兼容很好。

5.互动模式：互动模式的支持，您可以从终端输入执行代码并获得结果的语言，互动的测试和调试代码片断。

6. 可移植：基于其开放源代码的特性，Python已经被移植（也就是使其工作）到许多平台。

7. 可扩展：如果你需要一段运行很快的关键代码，或者是想要编写一些不愿开放的算法，你可以使用C或C++完成那部分程序，然后从你的Python程序中调用。

8. 数据库：Python提供所有主要的商业数据库的接口。

9. GUI编程：Python支持GUI可以创建和移植到许多系统调用。

10. 可嵌入：你可以将Python嵌入到C/C++程序，让你的程序的用户获得“脚本化”的能力。

## Python版本说明(以树莓派已安装版本为例)

	版本	说明	备注
	Python	Python2.7	属于Python早期版本，官方逐渐放弃维护该版本
	Python3	Pyhton3.7	Python3是较新的版本，Python2.7语法与Python3不兼容，其程序代码无法使用Python3的解释器运行

树莓派上运行python2环境，如下图：

A screenshot of a terminal window titled "192.168.0.101 - 远程桌面连接". The window title bar also shows "pi@raspberrypi: ~". The terminal prompt is "pi@raspberrypi:~ \$". The output shows:

```
pi@raspberrypi:~ $ python2
Python 2.7.16 (default, Oct 10 2019, 22:02:15)
[GCC 8.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

树莓派上运行python3环境，如下图：

A screenshot of a terminal window titled "192.168.0.101 - 远程桌面连接". The window title bar also shows "pi@raspberrypi: ~". The terminal prompt is "pi@raspberrypi:~ \$". The output shows:

```
pi@raspberrypi:~ $ python3
Python 3.4.3 |Continuum Analytics, Inc.| (default, Aug 21 2015, 00:53:08)
[GCC 4.6.3] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 555*666
369630
>>> 
```

## Python文件的执行

我们一般在终端里，通过 `python test.py` 的方式来运行Python文件。

## Python包管理工具—pip

pip是一个现代的，通用的Python包管理工具，提供了对Python包的查找，下载，安装，卸载的功能。

PIP的使用方法：

【注】：由于Python2与Python3的语法不兼容，故pip为Python2的包管理工具，Python3的包管理工具为“pip3”。 |pip命令|功能|备注|:-:|-:|-:|install|安装软件包|pip install [package name]| |download|下载软件包|| |uninstall|卸载软件包|pip uninstall [package name]| |list|列表列出已安装的软件包|pip list| |show|显示已安装软件包的信息|pip show [package name]| |check|检查已安装的软件包是否具有兼容的依赖项|| |search|搜索PyPI查找软件包|| |help|显示帮助命令||

## Python基本语法

The screenshot shows a Python script named `coffeeghost-q-in-py.py` in the UliPad 4.0 code editor. The code includes annotations explaining various Python concepts:

- 脚本文件一般用`.py`后缀
- 单行注释
- 导入其它代码模块
- 注意！Python最好也最个性的语法：使用缩进来代替其它语句声明；一般建议每个语句用4个空格来缩进。
- 变量得先实例化才可进一步计算
- 单行的话块，其实可以不换行的，但是，建议清晰起见，规范点：
  - 另起一行
  - 缩进一级
- 函数声明，注意使用冒号结束声明
- 多行注释的内容不用遵守当前缩进，只要开始的```缩进正确就成！
- 每级语块末尾```之类的括号引领！直接回车+4空格（当然，要在当前缩进基础上）
- 中文用户一定得先用这行来声明编码，同时文件本身也得存储成UTF-8编码！
- 给程序员的超快速Python脚本解释器，其实导入了`os.py`
- 函数名“main”在这儿并不是必须的，调用在这段脚本的最后部分；声明单行字符串，使用双/单引号都成，注意对字符串中的引号进行逃逸处理！
- print 这是Alice`的问候。 print 这是Bob`的问候。
- foo(5, 10) 字符可累，等于`'\*\*\*\*\*'
- counter = 0 连接字符串
- counter += 1 内置的列表类型对象，其实可以包含不同类型的数据，甚至可以包含其它列表对象；
- food = ['苹果', '杏子', '李子', '梨'] for i in food: print '数到10' range() 内置函数，返回类似 [0,1,2,3,4,5,6,7,8,9] 的数字列表，注意 for in 循环语句使用冒号结束声明！
- for i in range(10): print i 在循环中，i 指代了列表中按顺序的每个“food”
- def foo(param1, secondParam): 字串的格式化输出基本类似C语言的 res = param1+secondParam print %s 加 s 等于 %s%(param1, secondParam, res)
- if res < 50: 判断式也基本和C语言的相同
- elif (res>=50) and ((param1==42) or (secondParam==24)): | 逻辑运算符，不使用&& 和 ||，使用直观的E文单词
- else: | 这是单行注释
- return res | 这是多行注释.....
- if name == '\_\_main\_\_': | 一般在根本不愿使用主函数 main(); 而且使用内置的运行脚本名来判定；当仅当我们直接运行当前脚本时，name 才为 main，这样当脚本被当作模块进行 import 导入时，并不运行 main() 所以，一般这里是进行测试代码空置的...
- 关于 UliPad 4.0 作者: Limodou (limodou@gmail.com) 如果你有任何问题请与我联系。 The UliPad project homepage The UliPad mailinglist The UliPad Snippets Site My Blog Contact me 确定

## 1. Python语言源程序模块的初识

一个Python程序可能由一个或多个模块组成。模块是程序的功能单元。Python模块的典型结构由：模块文档、模块导入、变量定义、类定义语句、函数定义语句、主程序等组成。

**模块文档：**模块文档使用三引号注释的形式，简要的介绍模块的功能以及重要全局变量的含义。

**模块导入：**导入需要调用的其他模块。模块只能被导入一次，被导入模块中的函数代码并不会被自动执行，只能被当前模块主动（显式）调用。

## 2. 基本词法单位、标识符/常量/运算符等构成规则与关键字

基本词法单位：常量、变量、关键字、运算符、表达式、函数、语句、类等。常

量：是指初始化（第一次赋予值）后保持固定不变的值。例如：1，

3.14, 'Hello!', False, 这是4个不同类型的常量。在Python中没有命名常量，通常用一个不改变值的变量代替。比如：PI=3.14 通常用于定义圆周率常量PI。

标识符：用于不同的词法单位，通俗的讲就是名字。标识符可以作为变量、函数、类的名字。合法的标识符必须遵守以下规则：

- 由一串字符组成，字符可以是任意字母、数字、下划线、汉字，但这些字符中的开头不能是数字。
- 不能与关键字同名。关键字也成为保留字，是被语言保护起来具有特殊含义的词，不能用于起名字。查看Python的语言的所有关键字（用Python自带的IDLE执行下面两句代码）

```
import keyword  
keyword.kwlist
```

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 26 2018, 23:26:24)  
[Clang 6.0 (clang-600.0.57)] on darwin  
Type "copyright", "credits" or "license()" for more information.  
>>> import keyword  
>>> keyword.kwlist  
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class',  
'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from',  
'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass',  
'raise', 'return', 'try', 'while', 'with', 'yield']  
>>>
```

- 标识符中唯一能够使用的标点符号是下划线，不能含有其他标点符号（包含：空格、括号、引号、逗号、斜线、反斜线、冒号、句号、问号等）。

正确的标识符：X、var1、FirstName、stu\_score、平均分2等 错误的标识符：stu-score、First Name、2平均分 变量：是指在运行的过程中值可以被修改的量。变量的名称除了必须符合标识符的构成规则外，要尽量遵循一些约定俗成的规范。以下划线开头的变量在Python中有特殊的含义，所以自定义名称时，一般不用下划线作为开头字符。此外，Python是严格区分大小写字母的。也就是说，Score和score会被认为是两个不同的名字。运算符：指常量/变量之间进行何种运算。表达式：由常量、变量加运算符构成。一个表达式可能包含多次多种运算，与数学表达式在形式上很接近。例如：1+2、2(x+y)、0<=a<=10等。函数：是相对独立的功能单位，可执行一定的任务。语句：是由函数、表达式调用组成的。另外，各种控制结构也属于语句，例如：if语句、for语句。\*类：是同一类事物的抽象。我们处理的数据都可以看做数据对象。Python是面向对象的程序设计语言，它是一个事物的静态特征（属性）和动态行为（方法）封装在一个结构里，称之为对象。

### 3.程序的书写格式与基本规则

缩进：使用缩进来区分代码块的级别。Python语言中没有采用花括号或begin...end等来分隔代码块，而是使用冒号和代码缩进来区分代码之间的层次。代码缩进是一种语法规则，错误的缩进可能导致代码的含义完全不同。如下2个代码块

```
x = 0
if x == 1:
    x = x + 1
    print("x的值等于%d", x)

y = 0
if y == 1:
    y = y + 1
    print("x的值等于%d", x)
```

建议使用在缩进代码前输入4个空格来表示代码缩进，不推荐其他数量的空格或使用制表符的方式来完成缩进。

**分号：**Python允许在行尾加分号，但不建议加分号，也不要将两条命令放在同一行中。建议每一条命令单独一行。

**长语句行：**除非遇到长的导入模块语句或者注释里的URL，建议不宜超过80个字符。对于超长语句，允许但不提倡使用反斜杠连接行，建议在需要的地方使用圆括号来连接行。例如：

- 不推荐写法

```
year1 = 2016
if year1 % 4 == 0 and year1 % 100 != 0 or \
    year1 % 400 == 0:
    print(year1,"是闰年! ")
else:
    print(year1,"不是闰年! ")
```

- 推荐写法

```
year2 = 2018
if (year2 % 4 == 0 and year2 % 100 != 0 or
    year2 % 400 == 0):
    print(year2,"是闰年! ")
else:
    print(year2,"不是闰年! ")
```

**括号：**不建议使用不必要的括号，除非用于实现行连，否则不要在返回语句或者条件语句中使用括号，例如：

```
if (x):      # x两侧的括号多余
    foo()

if not (x):   # x两侧的括号多余
    foo()

return (x)     # x两侧的括号多余
```

空行：变量定义、类定义以及函数定义之间可以空两行。类内部的方法定义之间，类定义与第一个方法之间，建议一行。函数或方法中，如果有必要，可以空一行。

空格：对于赋值(=)、比较 (==,<,>,!=,<>,<=,>=,in,not in,is,is not)、布尔 (and,or,not) 等运算符，在运算符两边各加一个空格，可以使代码更清晰。而对于算数运算符，可以按照自己的习惯决定，但建议运算符两侧保持一致。例如：

- 不推荐写法

```
x==1
```

- 推荐写法

```
x == 1
```

注释：注释通常以#开始直到行尾结束。行内注释：和语句在同一行中的注释。行内注释应该以#和单个空格开始，应该至少用两个空格和前面的语句分开。注释块后面通常跟着代码，且注释块应该与相关代码的缩进一致。注释块中的每行以#和一个空格开始，注释块内段落以仅含单个#的行分割。注释块上下方最好各空一行。例如：

- 建议的写法

```
# 这个函数用于计算班级所有学生的平均分
#
# 例子: Avg(score,100)

def Avg(Score,Num):
    pass
```

文档字符串：是Python 语言独特的注释方式。文档字符串是包、模块、类或函数中的第一条语句。文档字符串可以通过对象**doc**成员被自动提取。我们书写文档字符串的时候，在其前、后使用三重双引号 """ 或三重单引号 ''。一个规范的文档字符串应该首先是一行概述，接着是一个空行，然后是文档字符串剩下的部分，并且应该与文档字符串的第一行的第一个引号对齐。例如：

```
def Avg(Score, Num=100):
    """
    计算班级的平均分

    从Score中读取所有学生的成绩，逐一加求总分，然后把总分除以人数Num，结果就是平均分，返回该

    参数
        Score: 记录所有学生的成绩列表
        Num: 班级总人数，默认值是100

    返回值
        float类型的平均分
    """
    pass
```

文档字符串可以通过**doc**成员进行查看，也可以在**help()**函数的结果里

```
>>> print(Avg.__doc__)
```

文档字符串通常用于提供在线帮助信息。

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-03-13

## 第2节 Linux基本操作

由于树莓派运行的 Raspbian 操作系统属于 Linux 家族，所以我们需要了解 Linux 的基本命令

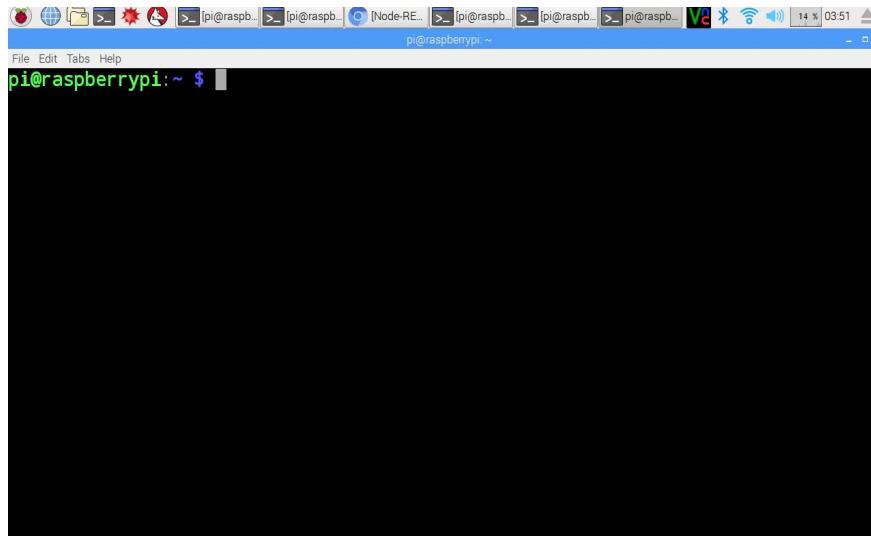
### 内容提要

- 了解Linux终端操作
- 体验Linux终端下的基本操作
- 学会Linux终端下的文件复制，移动，粘贴
- 学会Linux终端下Nano和Vim文件编辑器的使用

### 背景知识

#### Linux 终端

终端（Terminal）也称终端设备，是计算机网络中处于网络最外围的设备，主要用于用户信息的输入以及处理结果的输出。在此处为Linux操作系统用于用户输入信息和输出信息的窗口，也是人机交流的窗口。（如下图所示：）



#### 如何打开树莓派的终端(Terminal)

- 1.任意时刻按下键盘上的 `Ctrl + Alt + t`
- 2.点击快速启动栏上的终端图标

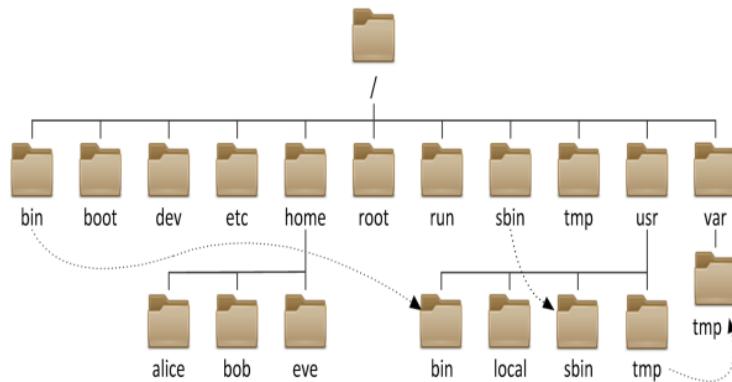
#### Shell

Shell 是一个程序，同时它又是一种程序设计语言。作为命令语言，它交互式解释和执行用户输入的命令或者自动地解释和执行预先设定好的一连串的命令；作为程序设计语言，它定义了各种变量和参数，并提供了许多在高级语言中才具有的控制结构，包括循环和分支。

在后边的学习中我们只会用到**Shell**命令，并且通过这些命令与树莓派(Linux)内核沟通，让树莓派执行我们想要的动作。

## Linux 目录结构

(如图所示：)



在Linux系统中，目录被组织成一个单根倒置树结构，文件系统从根目录开始，用/来表示。文件名称区分大小写（大小写敏感还需要看具体的文件系统格式），以.开头的为隐藏文件，路径用/来进行分割（windows中使用\来分割），文件有两个种类：元数据与数据本身。在操作linux系统时，通常会遵循以下的分层结构规则：

目录名称	功能
/	根目录，位于Linux文件系统目录结构的顶层，一般根目录下只存放目录，不要存放文件
/bin	提供用户使用的基本命令，存放二进制命令，不允许关联到独立分区
/boot	用于存放引导文件,内核文件,引导加载器
/sbin	管理类的基本命令，不能关联到独立分区，OS启动时会用到的程序
/lib	存放系统在启动时依赖的基本共享库文件以及内核模块文件
/lib64	存放64位系统上的辅助共享库文件
/etc	系统配置文件存放的目录，该目录存放系统的大部分配置文件和子目录，不建议在此目录下存放可执行文件
/home	普通用户主目录，当新建账户时，都会分配在此，建议单独分区，并分配额外空间用于存储数据
/root	系统管理员root的宿主目录
/media	便携式移动设备挂载点目录
/mnt	临时文件系统挂载点
/dev	设备（device）文件目录，存放linux系统下的设备文件，访问该目录下某个文件，相当于访问某个设备，存放连接到计算机上的设备
/opt	第三方应用程序的安装位置
/srv	服务启动之后需要访问的数据目录，存放系统上运行的服务用到的数据
/tmp	存储临时文件，任何人都可以访问,重要数据一定不要放在此目录下
/usr	应用程序存放目录，/usr/bin 存放保证系统拥有完整功能而提供的应用程序，/usr/share 存放共享数据，/usr/lib 存放不能直接运行的，却是许多程序运行所必需的一些函数库文件，/usr/local 存放软件升级包，第三方应用程序的安装位置，/usr/share/doc 系统说明文件存放目录
/var	放置系统中经常要发生变化的文件，如日志文件
/proc	用于输出内核与进程信息相关的虚拟文件系统，目录中的数据都在内存中
/sys	用于输出当前系统上硬件设备相关的虚拟文件系统
/selinux	存放selinux相关的信息安全策略等信息

## 常用的Shell命令

### 磁盘操作命令

注：命令在输入不完整时可食用 **Tab** 键补齐命令或路径上存在的文件名

1. **ls** 功能: 用于显示指定工作目录下之内容 (列出目前工作目录所含之文件及子目录) 格式: `ls [-参数] [名称]` 参数: `-a` 显示所有文件及目录  
`-l` 显示文件或文件夹的详细信息 实例: `ls -al` 列出当前目录下所有文件(包括隐藏文件)的详细信息 `ll` 显示文件或文件夹的详细信息(系统内置的快捷指令, 相当于 `ls -l`)
2. **cd** 功能: 用于切换当前工作目录至指定目录 格式: `cd [指定目录]`  
参数: 无 实例: `cd ~` (~ 指 `/home/user/` 目录)进入 `home` 目录(进入 `home` 目录也可以直接 `cd + Enter`) `cd /` 进入根目录 `cd 123`  
进入当前目录下的 `123` 文件夹 `cd ../123` 进入上级目录下的 `123` 文件夹
3. **cp** 功能: 用于复制文件或目录 格式: `cp [参数] 源文件 目标目录` 参数: `-a` 此选项通常在复制目录时使用, 它保留链接、文件属性, 并复制目录下的所有内容 `-r` 若给出的源文件是一个目录文件, 此时将复制该目录下所有的子目录和文件 实例: `cp 123.c 123.c.back` 将当前目录下的 `123.c` 文件复制并重命名为 `123.c.back` 到当前文件夹 `cp ..../123.c ./` 复制上级目录下的 `123.c` 文件到当前目录下 `cp -r ..//dir1/ ./` 复制上级目录下的 `dir1` 文件夹及其所有文件到当前目录下
4. **touch** 功能: 用于修改文件或者目录的时间属性, 若文件不存在, 系统会建立一个新的文件 格式: `touch [文件名]` 参数: 无 实例: `touch 123.txt` 在当前目录下创建 `123.txt` 文件
5. **mkdir** 功能: 用于创建文件夹 格式: `mkdir 文件夹名称` 参数: `-p` 建立多级文件夹 实例: `mkdir dir1` 在当前目录下创建名称为 `dir1` 的文件夹 `mkdir -p dir1/dir2/dir3/` 在当前目录下连续创建 `dir1`, `dir2`, `dir3` 的子目录
6. **mv** 功能: 用来为文件或目录改名、或将文件或目录移入其它位置 格式: `mv [参数] 源文件 目标文件` 参数: `-i` 若指定目录已有同名文件, 则先询问是否覆盖旧文件 `-f` 在 **mv** 操作要覆盖某已有的目标文件时不给任何指示; 实例: `mv 123.c 456.c` 将当前目录下的 `123.c` 文件重命名为 `456.c` `mv 123.c ./dir1/` 将当前目录下的 `123.c` 文件移动到当前目录的子目录 `dir1` 下
7. **rm** 功能: 用于删除一个文件或者目录 格式: `rm [参数] 名称` 参数: `-r` 将目录以及目录中的文件全部删除 `-f` 即使文件属性为只读, 也会直接删除, 无需逐一确认 实例: `rm 123.c` 将当前目录下的 `123.c` 文件删除(无确认提示, 无法恢复) `rm -rf ./dir1/` 删除当前目录下的 `dir1` 子目录及其内所有文件, 且无需确认, 直接删除, 不可恢复
8. **cat** 功能: 用于查看文件的内容(将文件内容打印到终端上用以查看) 格式: `cat 目录+文件` 参数: 无 实例: `cat 123.c` 查看 `123.c` 文件的内容
9. **pwd** 功能: 用于显示当前工作目录的绝对路径 使用方法: 在终端键入: `pwd`, 按 `Enter` 键, 即显示当前工作目录的绝对路径

## 系统操作命令

1. Linux 操作系统命令 命令名称 | 执行动作 | 备注 :|:|:-: sudo apt-get update | 更新软件源|由于改变系统环境, 故加 `sudo` `sudo apt-get upgrade` |更新已安装的软件版本|由于改变系统环境, 故加 `sudo` `sudo apt-get dist-upgrade` |更新系统|由于改变系统环境, 故加 `sudo` `sudo apt-get install [软件名]` |安装软件|由于改变系统环

境，故加 `sudo sudo apt-get remove [软件名]` |卸载软件|由于改变系统环境，故加 `sudo sudo apt-get autoremove` |自动卸载不需要的软件包|由于改变系统环境，故加 `sudo apt-cache search [软件名]` |搜索指定名称的软件包|由于未对系统做出改变，不需加 `sudo apt-cache show [软件名]` |获取包的相关信息，如说明、大小、版本|由于未对系统做出改变，不需加 `sudo apt-get source [软件名]` |下载软件包的源代码|由于未对系统做出改变，不需加 `sudo sudo apt-get clean` |清理无用软件包|与下一条命令配合使用 `sudo apt-get autoclean` |清理无用软件包|与上一条命令配合使用 `whereis [软件名]` |搜索软件并得到其详细信息| `which [命令]` |在系统中搜索命令以确定该命令是否存在|

2. `sudo` 功能：以超级用户身份运行软件或修改系统文件(类似windows10上的 以管理员身份运行 ) 实例：`sudo apt-get install nano` :安装 nano 文本编辑软件时，由于是更改系统环境，所以需要加 `sudo`

3. `sudo shutdown -r now` 重启系统    `sudo shutdown -h now` 关机

4. `chmod` 功能：Linux的文件调用权限分为三级：文件所有者、所属组、其他。`chmod` 可修改文件的使用权限，即是否允许文件被其所有者，所属组，其他进行 读(r)，写(w)，执行(x) ， 三类权限分别对应八进制数： r -> 4 , w -> 2 , x -> 1  
说明：(1) User (所有者) , Group(所属组), Other(其他)分别都拥有对该文件的 rwx 权限，即： 文件所属 | User | Group | Other :|-:|-:|-:|:- 权限|rwx|rwx|rwx 各权限值|421|421|421 八进制值|7|7|7 实例：`sudo chmod 666 123.sh` 去除当前目录下 `123.sh`文件的所有所属的 可执行权限x (八进制数为 1 )        `sudo chmod 775 123.sh` 为当前目录下 `123.sh`文件的所有所属添加 可执行权限x

5. `chown` 功能：指定文件的拥有者改为指定的用户或组，用户可以是用户名或者用户ID；组可以是组名或者组ID 实例：`sudo chown root:root 123.py` 将 `123.py`的用户以及用户组改为 `root` (管理员用户) 实例：`sudo chown pi:pi 123.py` 将 `123.py`的用户以及用户组改为 `pi` (普通用户)

6. `netstat` 功能：用于显示与IP、TCP、UDP和ICMP协议相关的统计数据，一般用于检验本机各端口的网络连接情况。（如下图所示）

```
[root@mysql ~]# netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      52 192.168.25.138:ssh      192.168.25.100:51068    ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type      State          Listen Address
unix  2          [ ]        DGRAM      8981      @/org/kernel/udev/udevd
unix  2          [ ]        DGRAM      12440     @/org/freedesktop/hal/udev_event
unix  9          [ ]        DGRAM      11690     /dev/log
unix  2          [ ]        DGRAM      13883
```

解析：(1) `netstat` 会显示当前 活动的网络连接(Active Internet connections) 和 活动的本地进程间通信的状态； (2) 通过查看这些信息我们能知道当前有哪些通信端口被占用以及当前程序工作的状态。

7. `pkill` 功能：控制(关闭)同名程序的所有进程 语法：`pkill 选项 pattern (模式)`  
参数：`-G`:仅匹配真实组ID在给定列表中的进程。                  `-t termlist`:仅匹配与给定列表中终端关联的进程。                  `-signal`:指定发往每一个匹配进程的信号  
`-U uidlist`:仅匹配真实的用户ID在给定列表中的进程。 实例：`pkill -9 -U UserName` 强制退出 `UserName` 用户

## 终端下常用应用软件的使用方法

1. **tar** 功能: 将单个或多个文件(文件夹)打包或打包并压缩或解压 参数: -c: 新建打包文件, 同 -v 一起使用 查看过程中打包文件名 -x: 解压文件

-C: 解压到对应的文件目录 -f: 后边接要处理的文件 -

j: 通过**bzip2**方式压缩或解压, 最后以**.tar.br2**为后缀。压缩后大小小于**.tar.gz**

-z: 通过**gzip**方式压缩或解压, 最后以**.tar.gz**为后缀 -v: 显示解

压或压缩的过程 -t: 查看打包文件中内容, 重点文件名 -u: 更新

压缩文件中的内容 -p: 保留绝对路径, 即允许备份数据中含有根目录

-P: 保留数据原来权限及属性 --exclude = FILE: 压缩过程中不

包含 FILE 文件 实例: (1) `tar czvf all-txt.tar.gz ./*.txt` 将当前目录下的所有文  
件名以 `.txt` 结尾的文件, 使用**gzip**方式压缩打包 `all-txt.tar.gz` 文件 (2)

`tar xvf JieYaWenJian.tar.gz` 将 `JieYaWenJian.tar.gz` 文件解压到当前文件夹下

2. **zip** 功能: `zip` 压缩文件 参数: -d: 从压缩文件内删除指定的文件 -

D: 压缩文件内不建立目录名称 -F: 尝试修复已损坏的压缩文件

-g: 将文件压缩后附加在既有的压缩文件之后, 而非另行建立新的压缩文

件 -i: <范本样式> 只压缩符合条件的文件 -m: 将文件压缩并加

入压缩文件后, 删除原始文件, 即把文件移到压缩文件中 -n: <字尾字符

串> 不压缩具有特定字尾字符串的文件 -o: 以压缩文件内拥有最新更改时

间的文件为准, 将压缩文件的更改时间设成和该文件相同 -r: 递归处理,

将指定目录下的所有文件和子目录一并处理 -q: 不显示指令执行过程

-V: 保存VMS操作系统的文件属性 实例: (1) `zip -q -r html.zip`

`/home/html` 将 `/home/html/` 这个目录下所有文件和文件夹打包为当前目录下的

`html.zip` (2) `zip -dv cp.zip a.c` 从压缩文件 `cp.zip` 中删除文件 `a.c`

3. **unzip** 解压 `*.zip` 文件(\* 为通配符, 代指任意字符) 用法: `unzip JieYaSuo.zip`

将 `JieYaSuo.zip` 文件解压到当前文件夹

4. **wget** 功能: 下载URL所连接的文件到本地 用法: `wget [参数] [URL]` (`URL` 即文  
件的下载链接, 可通过浏览器点击鼠标右键并选择 复制链接地址 来获取目标的URL)

参数: -v: 显示Wget的版本信息并退出 -b: wget启动后转入后台

-q: 安静模式(无信息输出) -v: 详尽的输出(此为默认值)

-nc: 不要重复下载已存在的文件 -c: 继续下载部分下载的文件

-O: 下载文件到对应目录, 并且修改文件名称 -spider: 模拟下

载, 不会下载, 只是会检查是否网站是否正常可用 实例: (1) `wget`

`http://mirrors.aliyun.com/ubuntu-16.04.6.iso` 将 `ubuntu-16.04.6.iso` 文件下载到当前目  
录 (2) `wget --spider www.baidu.com` 模拟下载, 不会下载, 只是会检查是否

网站是否正常可用

5. **curl** 功能: curl是一个非常实用的、用来与服务器之间传输数据的工具 用

法: (1) `curl http://localhost:8080/simple-service-webapp/test/hello` 获取页面内容

```
C:\tools\2019-professional\cmder
λ curl http://localhost:8080/simple-service-webapp/test/hello
hello
```

(2) `curl -I http://localhost:8080/simple-service-webapp/test/hello` 显示

HTTP头, 而不显示文件内容, 使用 -I 选项

```
C:\tools\2019-professional\cmder
λ curl -I http://localhost:8080/simple-service-webapp/test/hello
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/plain
Content-Length: 5
Date: Sun, 26 May 2019 13:04:51 GMT
```

```
(3) curl -o save.txt http://localhost:8080/simple-service-webapp/test/hello
```

将返回的结果保存到文件

```
λ curl -o save.txt http://localhost:8080/simple-service-webapp/test/hello
  % Total    % Received % Xferd  Average Speed   Time   Time     Time  Current
          Dload  Upload Total   Spent    Left Speed
100 保  5 100  5    0      0  106      0 --:--:-- --:--:-- --:--:-- 106

C:\tools\2019-professional\cmder
λ cat save.txt
hello
```

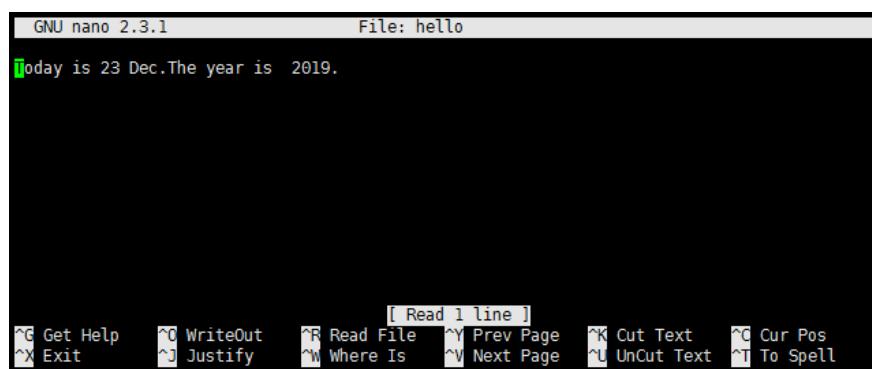
6. `git clone` 功能：将URL指定的github代码仓库克隆到本地当前目录 用法：`git clone [git仓库的URL]` 实例：`git clone https://github.com/esp8266/Arduino.git` 将此链接对应的仓库克隆到本地当前目录

## 使用文本编辑器 nano 编辑文件

1. 使用nano文本编辑器打开文件 `123.txt`：`nano 123.txt`

2. 打开文件的同时显示文件行数 `nano -c 123.txt`

3. 若编辑文件时提示“权限不够”，则在命令前加 `sudo` 即：`sudo nano 123.txt`



4. 保存使用nano编辑好的文件：键入 `Ctrl + o`, 然后按下 `Enter` 键 退出nano编辑器：键入 `Ctrl + x`

## 其他常见Linux命令

命令名称	执行动作	备注
<code>Alt + Tab 键</code>	切换活动窗口	
<code>Tab 键</code>	自动补齐	
<code>man [命令]</code>	查看命令使用手册	
<code>ifconfig</code>	查看当前网路连接状态以及IP地址	
<code>ping 「IP地址」/[URL]</code>	检测与某个IP地址是否连通	
<code>sudo raspi-config</code>	打开树莓派配置界面	只针对树莓派系统
<code>date</code>	查看当前系统时间	
<code>date 011318172020.00</code>	设置时间，格式为月日时分年.秒	
<code>ps -ax</code>	显示当前运行的进程	
<code>kill -9 [进程号]</code>	关闭某个进程	
<code>top</code>	实时显示各个进程对资源的占用情况	
<code>passwd</code>	设置用户密码	
<code>groups</code>	显示当前用户所属组	
<code>clear</code>	清空终端屏幕	
<code>uname -m</code>	显示机器的处理器架构	
<code>which [命令]</code>	在系统中搜索命令以确定该命令是否存在	

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-02-26

## 第3节 Web服务基础

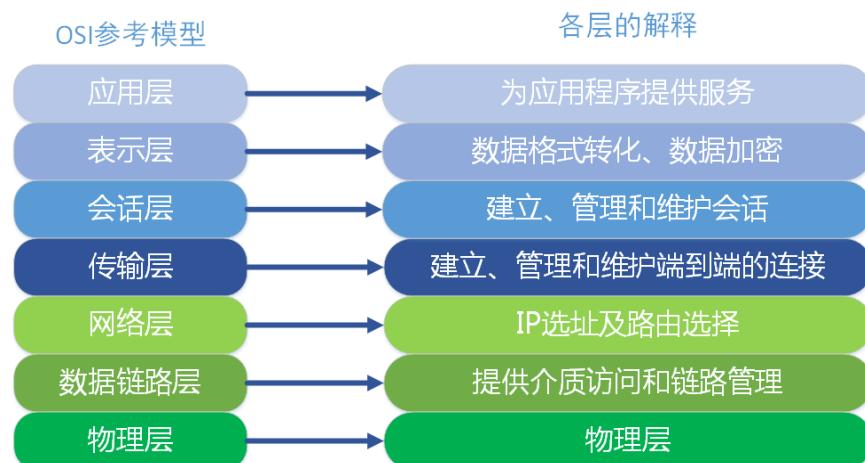
### OSI七层网络模型

我们在后面的章节中会接触到很多和网络相关的内容，比如第三章的小车控制，是开发板作为服务器，使用手机等访问服务器，对小车进行运动控制，接收服务器传来的小车摄像头画面等。第六章通过物联网平台，控制机器人的姿态、语音交互、各种传感器数据的读取和处理等。

OSI（Open System Interconnect），即开放式系统互联。一般都叫OSI参考模型，是ISO（国际标准化组织）组织在1985年研究的网络互连模型。是互联网最基本也是重要的知识。ISO为了更好的使网络应用更为普及，推出了OSI参考模型。其含义就是推荐所有公司使用这个规范来控制网络。这样所有公司都有相同的规范，就能互联了。

### OSI七层模型的划分

OSI定义了网络互连的七层框架（物理层、数据链路层、网络层、传输层、会话层、表示层、应用层），即ISO开放互连系统参考模型。如下图。



每一层实现各自的功能和协议，并完成与相邻层的接口通信。OSI的服务定义详细说明了各层所提供的服务。某一层的服务就是该层及其下各层的一种能力，它通过接口提供给更高一层。各层所提供的服务与这些服务是怎么实现的无关。

### 传输层协议：TCP、UDP

顾名思义，传输层主要的功能是传递信息。传输层建立了主机端到端的链接，我们通常说的，TCP UDP就是在这一层。端口号即是这里的“端”。端是由应用层来决定的。

我们在后面章节接触到的机器人或小车，我们也是通过TCP协议，通过访问IP地址和端口号，来与它们交换信息的。

### 应用层协议：HTTP、FTP、SMB

网络中的计算机是通过IP地址来代表其身份的，IP地址（公网IP）能表示某台特定的计算机，但是一台计算机上可以同时提供很多个服务，如数据库服务、FTP服务、Web服务等，我们就通过端口号来区别相同计算机所提供的这些不同的服务，如常见的端口号21表示的是FTP服务，端口号23表示的是Telnet服务端口。端口号80和443是http常用的端口。同学们可以在浏览器的地址栏尝试输入<\_0 class="copyright">、<\_43 class="copyright">，观察有无区别。一般来说，网址或IP地址后面如果不输入特定端口，默认是80端口。

## Web服务器Apache与Nginx

Apache是Apache软件基金会下的一个项目—Apache HTTP Server Project，Nginx同样也是一款开源的HTTP服务器软件。HTTP服务器软件本质上也是一种应用程序——它通常运行在服务器之上，绑定服务器的IP地址并监听某一个端口来接收并处理HTTP请求，这样客户端（一般来说是IE, Firefox, Chrome这样的浏览器）就能够通过HTTP协议来获取服务器上的网页、文档、音频、视频等等资源。

## 安装与配置Nginx

我们后面使用的树莓派，运行的是Linux的一个重要的发行版Debian。在Debian操作系统，通过终端可以很方便地安装和部署Nginx服务器。

### 安装Nginx

1. 打开终端
2. 运行命令 `sudo apt install nginx`
3. 安装完毕后，运行 `sudo systemctl enable nginx`，`sudo systemctl start nginx`
4. 打开浏览器，输入树莓派的IP地址，看看是不是打开了一个Nginx的说明呢，这就表示我们安装成功了。其他同学可以通过访问你的树莓派IP地址，来看到你发布的内容了。网站的搭建是不是非常简单呢？

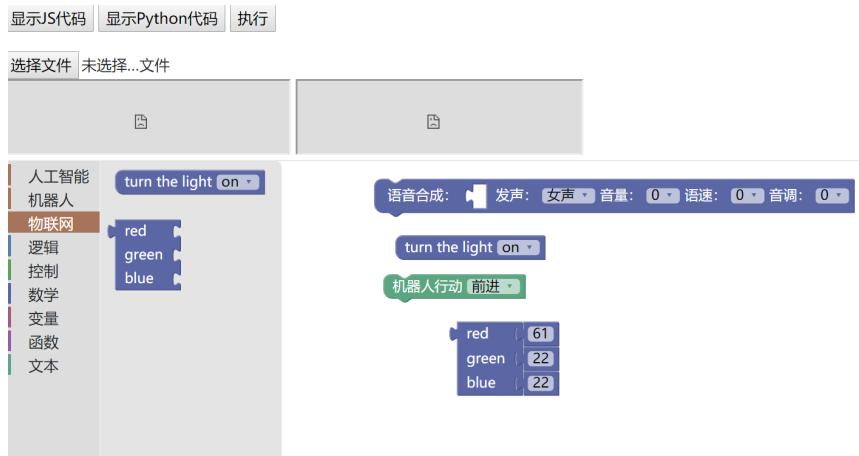
## 使用WordPress或Typecho，发布个人网站

同学们如果对搭建个人网站感兴趣，可以了解一下以下三个网站系统，WordPress是基于PHP和MySQL的一个人个人博客网站系统，Typecho是一个较为轻量级的个人博客网站系统。

## Scratch与Blockly



这只可爱的小猫就是Scratch的吉祥物。Blockly和Scratch都是开源的网络程序。Scratch比Blockly更早诞生，到了Scratch 3.0，Scratch开始使用Blockly进行构建。我们可以比像部署WordPress或Typecho更容易来在服务器上部署Blockly和Scratch。



上图是Blockly的一个Demo。通过它，我们可以用积木的方式来控制小车和机器人、灯、甚至是电视机和空调。在后面的学习中，我们将深入地了解它们。

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-03-01

# 开源硬件基本操作

## 什么是“开源”

**Open Source Code(开放源代码):** 开放源代码（Open source code）也称为源代码公开，指的是一种软件发布模式。一般的软件仅可取得已经过编译的二进制可执行档，通常只有软件的作者或著作权所有者等拥有程序的原始码。有些软件的作者会将原始码公开，被公开的原始程序代码称为“开放源代码”。

开放源代码软件源于自由软件开源运动，简称开源软件。是指那些源代码公开，可以被自由使用、复制、修改和再发布的一系列软件的集合。由此得出开源软件的几个特点，即：

1. 代码自由使用并可再发行
2. 开源软件发行时其源代码要一并发行
3. 允许他人对既有的源码进行修改并再次发布
4. 原始创作者保证源代码的完整性
5. 不歧视程序在任何领域内的使用
6. 基于源程序的新产品也要遵循同样的“开源许可协议”等

## 开源许可协议

自由软件/开源软件是自由的，免费的，源代码开放的，可自由下载安装和使用。同时，为了维护作者和贡献者的合法权利，保证这些软件不被一些商业机构或个人窃取，影响软件的发展，开源社区开发出了各种的开源许可协议，如：

**GPL协议, COPYLEFT协议, LGPL协议, Apache License协议, BSD协议**

重点介绍**GPL协议**，其授予程序接受人以下权利(自由)：

1. 以任何目的运行此程序的自由
2. 以学习程序工作机理为目的，对程序进行修改的自由(能得到源代码是前提)
3. 再发行复印件的自由
4. 改进此程序，并公开发布改进的自由(能得到源代码是前提)

例如：全球所有搭载**Android**操作系统的手机，其操作系统部分要遵循一定的开源协议，因为**Android**操作系统的内核是**Linux**，而**Linux**正是基于开源许可协议-GPL协议的操作系统，所以，以它为核心的**Android**也要遵循同样的**GPL**协议。如下

图：



## 开源硬件的分类

### 主控板

开源硬件中有一部分根据其功能作用，称之为“主控板”。而主控板上的核心正是能够运行程序代码的“小电脑”，它能将我们的想法，也就是程序转换为其上引脚电信号的变化，通过这些变化的电信号来达成我们的想法。

### 外设和传感器

本次AI课程配有：

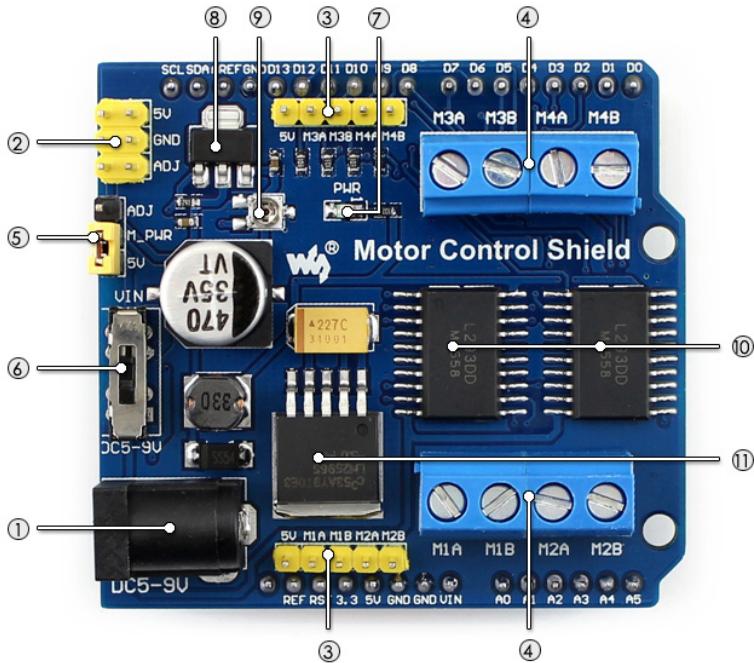
外设：舵机，舵机云台，TT电机，TT电机小车

传感器：超声波测距传感器，温湿度传感器等

## 开源硬件的基本操作

### 扩展板

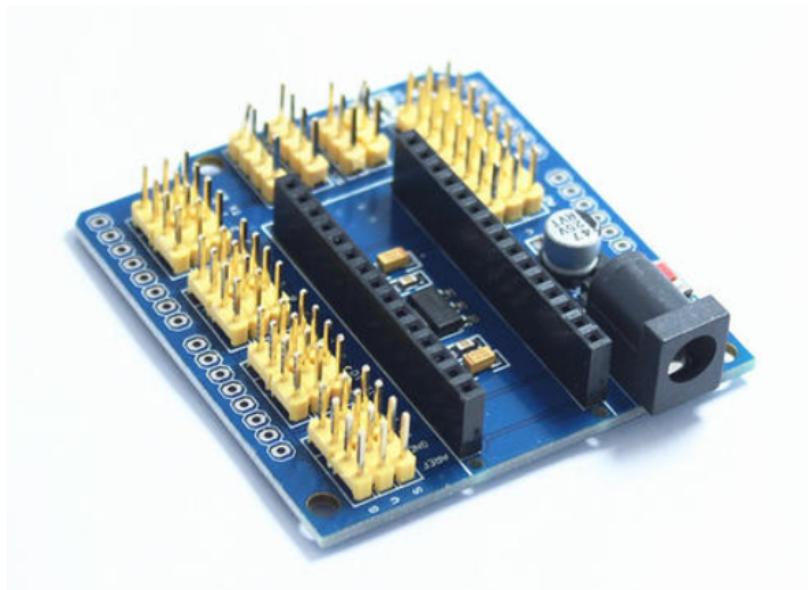
#### 1. Arduino 电机扩展板



此扩展板搭载了两颗L293DD电机驱动芯片和独立的稳压电路，可通过DC插头给该扩展板提供外部5V-12V电源，2颗L293DD电机驱动芯片可驱动4个直流电机，为小车平台提供了驱动基础。

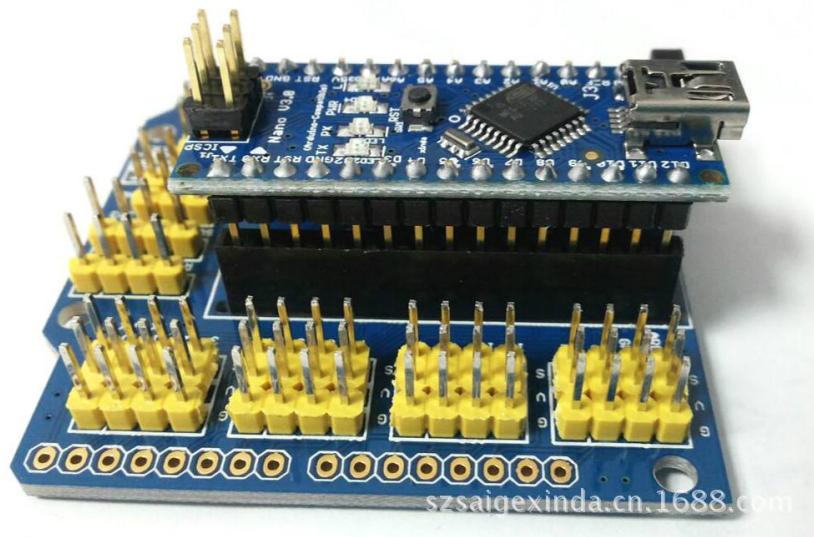
由于直流电机可通过电流的方向来改变转向，即可通过互换直流电机的两根引线来改变电机的运转方向。

## 2. arduino nano扩展板

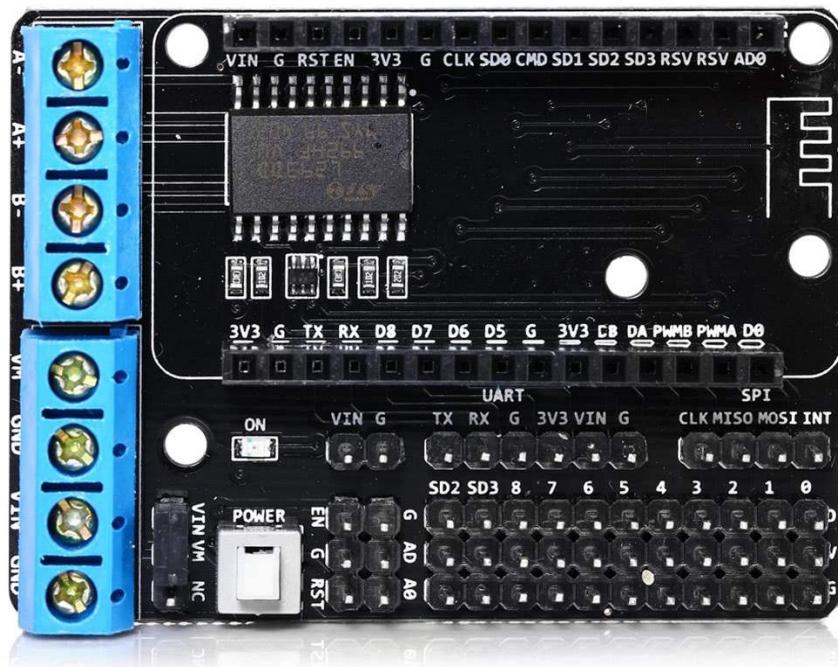


此扩展板为引脚和电源扩展板，即把体积较小的Arduino Nano的所有引脚都印出来，并对每个引脚都配有供电引脚(VCC和GND)，同时配有5mmDC接口，可接7-12V电源给arduino nano供电，且每个数字引脚都符合“舵机”控制线的定义，可以直

接接入舵机。此扩展板与Arduino Nano的组合状态如下图：

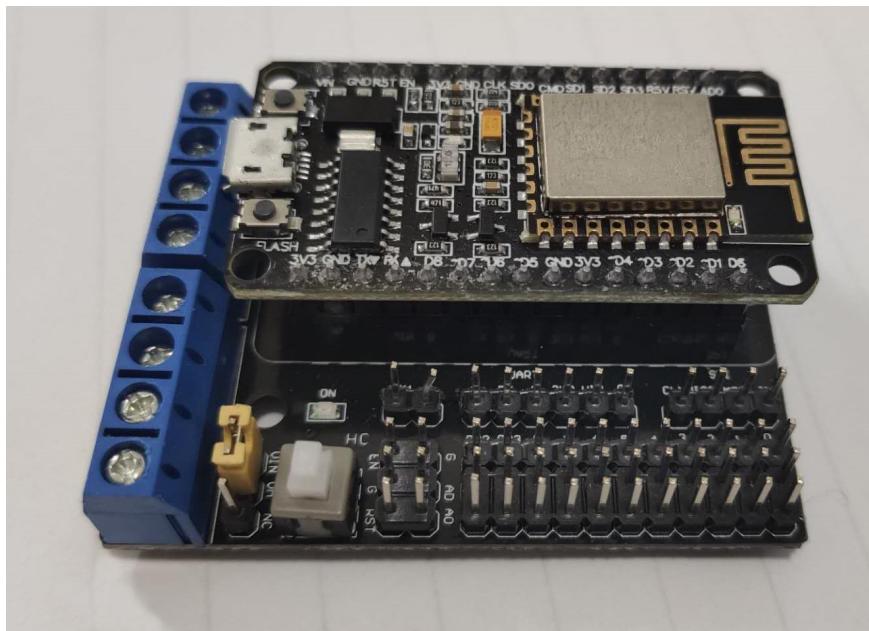


### 3. NodeMcu 电机扩展板

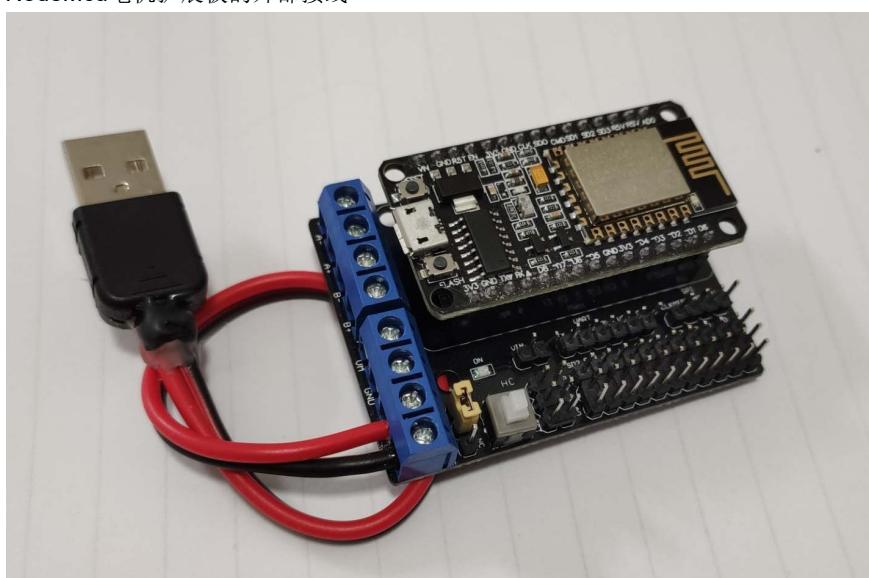


此电机扩展板集成一颗L293DD直流电机驱动芯片，可以同时驱动两路直流电机，同时将NodeMcu开发板的所有引脚印出来，并对每个引脚配有电源(VCC和GND)插针，可直接入舵机，并可为需要5V或3.3V电源的设备供电，同时也可通过VIN接线端子输入5V电源，Power开关可控制外部电源的通断。

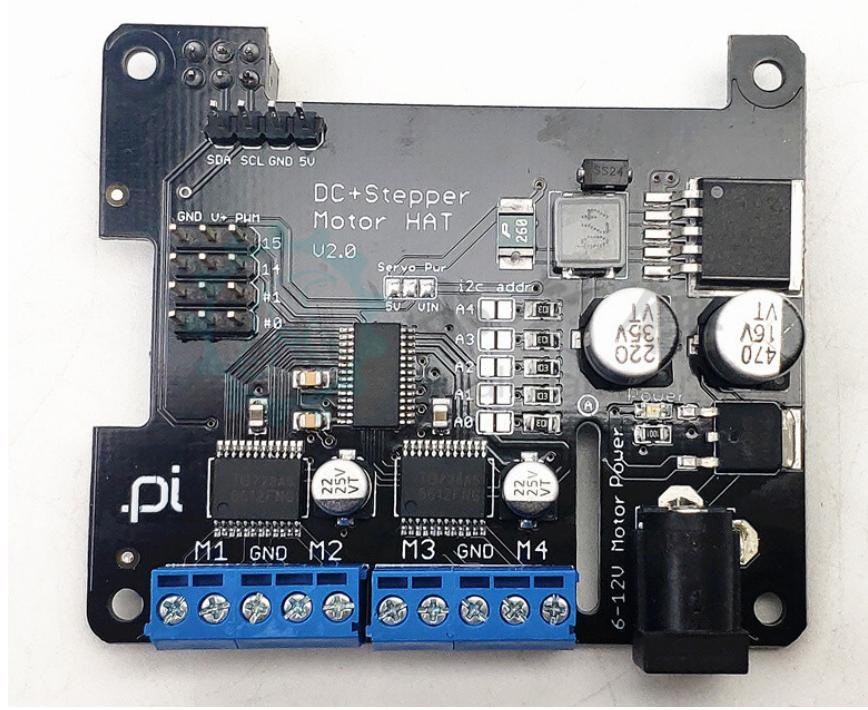
NodeMcu与此扩展板组合状态如下图所示：



NodeMcu电机扩展板的外部接线



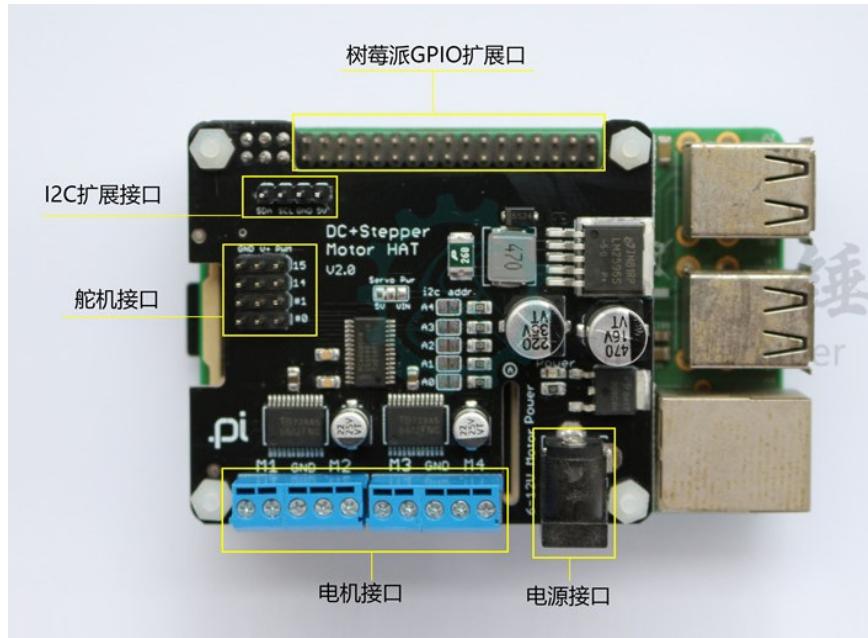
#### 4. 树莓派电机扩展板



此电机扩展板通过I2C接口接入树莓派，通过I2C总线芯片外挂两颗直流电机驱动芯片，可同时驱动4个直流电机，同时又引出了单独的一组I2C总线接口，除此之外，此扩展板带有4个标准舵机PWM输出接口，可以控制4路舵机。

由于该扩展板为大功率输出板，所以板上带有独立外部供电接口，使用此驱动板控制电机时需要外接5V-12V电源，否则不能正常工作。

此电机扩展板与树莓派组合状态如下图：

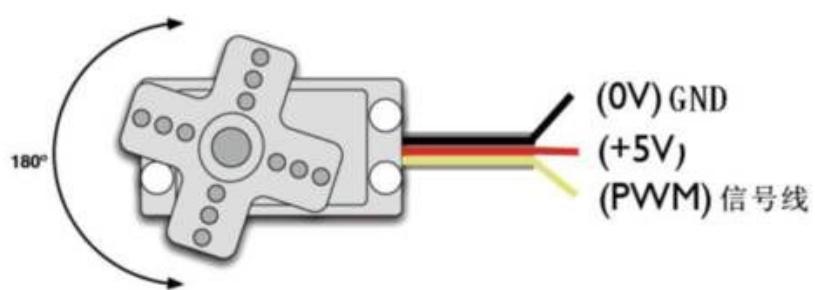


## 外设和传感器

### 1. 舵机



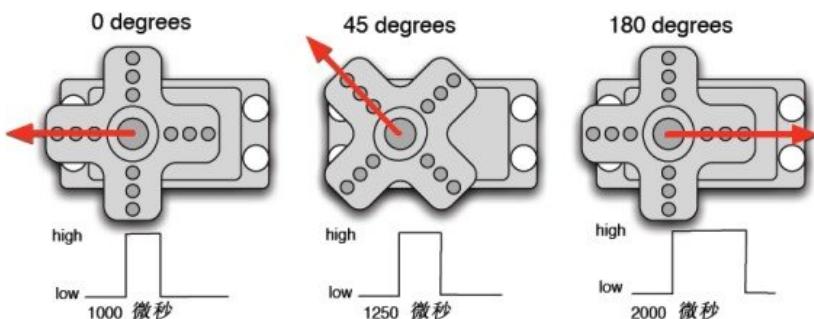
如上图所示,舵机是一种由直流电动机,减速齿轮组,角度控制器和动力输出轴组成的一种动力提供机械,主要作用是根据控制信号使动力输出轴转动一定角度。通常使用舵机来控制一些车船模型,以及机器人等。



舵机有3条线,其定义见下表:

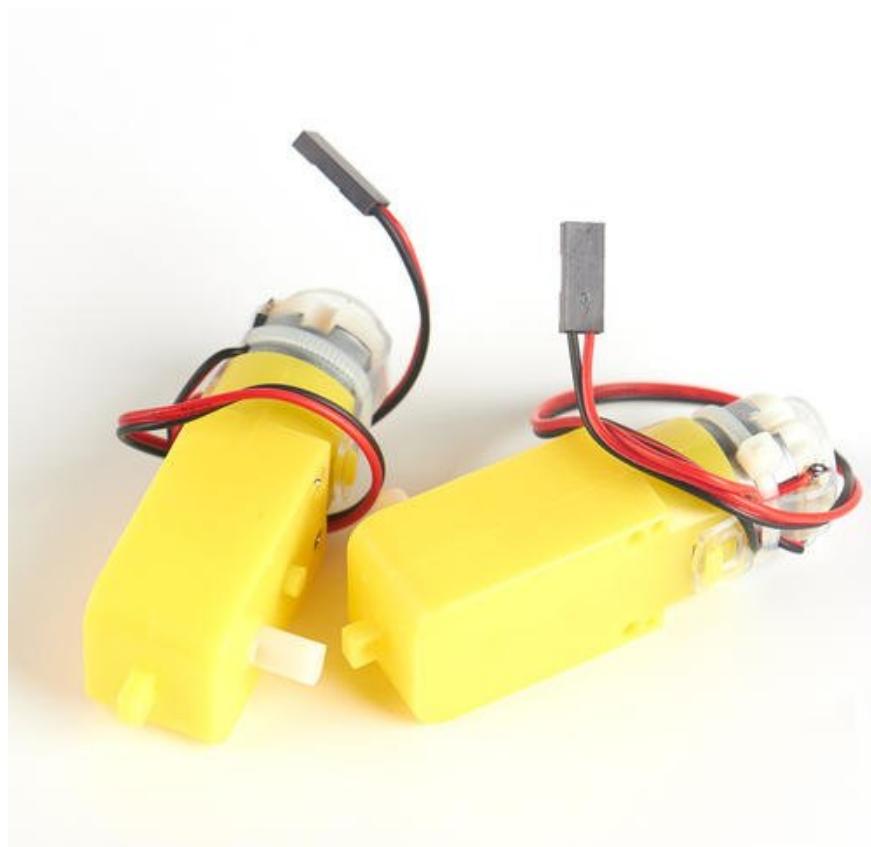
|颜色|作用| :-:-:  
|黄色|控制信号(PWM)输入线| |红色|5V供电线(电源正极)| |GND|电源负极|

注: 第一根黄色线为信号线,需要为其输入PWM(脉冲宽度调制)信号才能使舵机正常工作。如下图所示:



即在一个脉冲循环内,高电平持续时间占总循环时间的比率(占空比)越大,舵机转动的角度就越大。

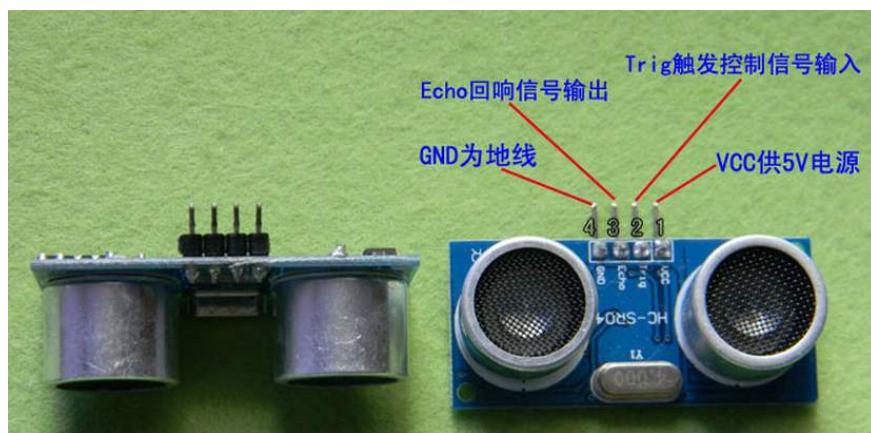
## 2. 小车直流电机



小车直流电机也叫TT电机，单个电机由一个直流马达和苏联减速齿轮构成，通过控制直流马达的正反转控制方向，控制直流马达的转速来控制速度。

在实际使用时可通过调换两条引出线在驱动板的位置来更改旋转方向。

## 3. 超声波测距传感器

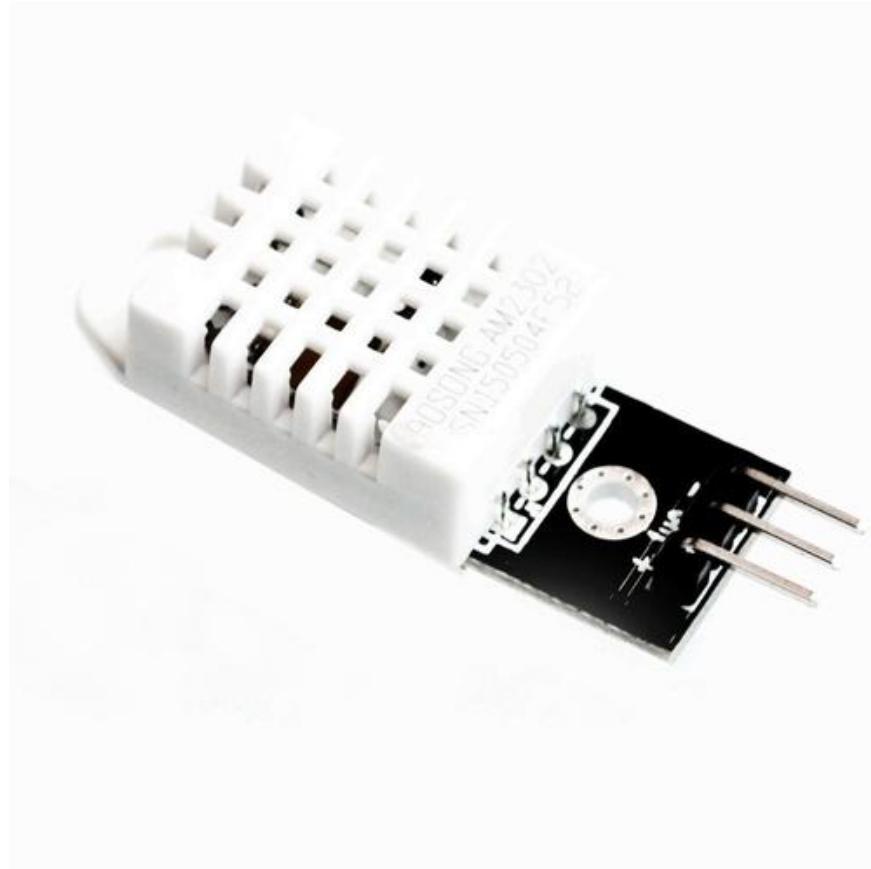


超声波测距传感器(如下图所示)通过发出超声波(频率为20000Hz的声波)，再被阻挡物体表面反射，反射的超声波又被接收装置接收，然后按下方照公式计算出来：  
距离 = 声速 × 发出超声波到被接收返回的时间 / 2

超声波测距传感器的接口有4根引脚，基本定义为：

|名称|作用| :-:-: | |VCC|5V供电接口(电源正极)| |Trig|触发控制信号输入| |Echo|回响信号输出| |GND|电源负极|

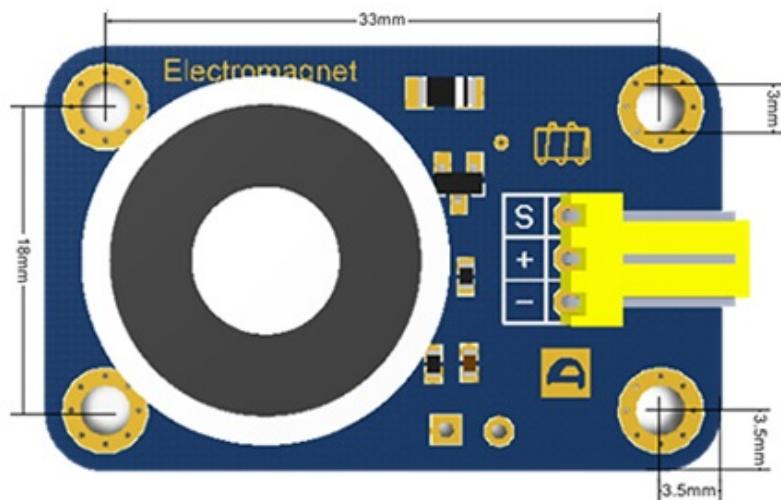
#### 4. DHT22温湿度传感器



DHT22温湿度传感器也称AM2302，是一款含有已校准数字信号输出的温湿度复合传感器，湿度量程范围0~99.9%RH，精度 $\pm 2\%$ RH，而温度量程范围是-40°C~80°C，精度 $\pm 0.5^\circ\text{C}$ 。DHT22传感器也是单总线传感器，即其输出的数据通过一个引脚即可输出，其接口定义如下表所示：

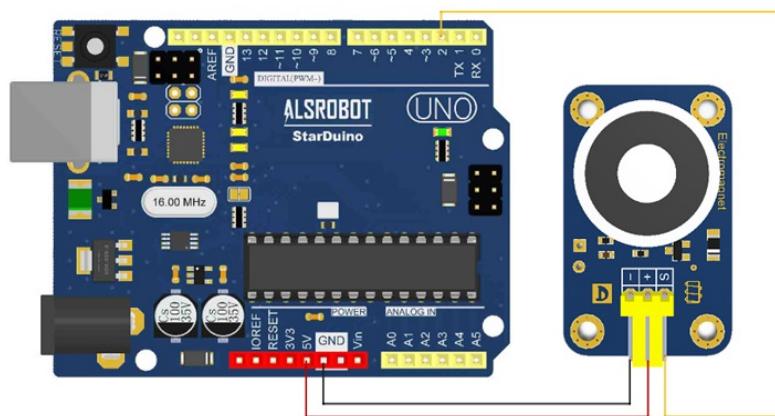
|名称|作用| :-:-: | |VCC(+)|5V供电接口(电源正极)| |out|信号输出引脚| |GND(-)|电源负极|

#### 5. 电磁铁



电磁铁是给绕在铁质内芯的线圈通电而产生磁力，而断电后磁力消失的装置。其控制接口定义如下表所示：|名称|作用|:-:-|S(signal)|开关控制信号：高电平产生磁力；低电平磁力消失|+|(5V)|电源正极|-(GND)|电源负极|以Arduino UNO为例，电磁铁的接线方法如下图：

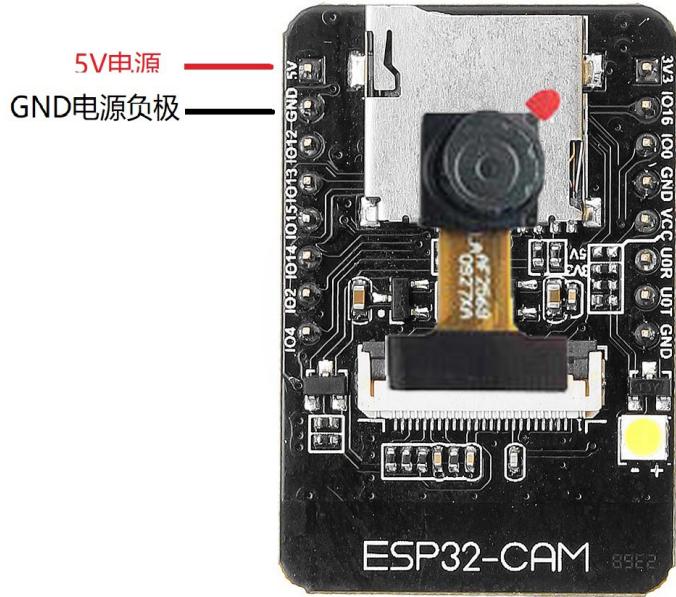
### 接线示意图



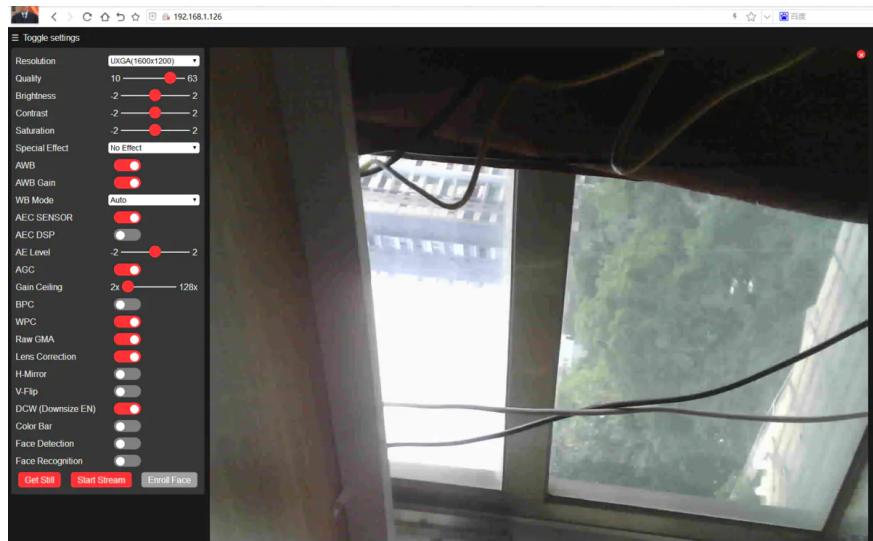
## 6. Esp32-Cam无线摄像头



ESP32-Cam是以ESP32芯片为计算核心，并搭载原生WiFi控制器的无线摄像头。它能够根据用户烧录的程序代码，使用户在连接它wifi名称的电脑网页端看到摄像头拍摄的实时视频，如下图所示：



由于此模块已提前烧录好程序，且其在小组局域网中的IP地址也已固定下来，所以在使用时只需按照下图连接电源线，稍等片刻，在同局域网电脑的浏览器中输入该摄像头的IP地址，按回车即可：

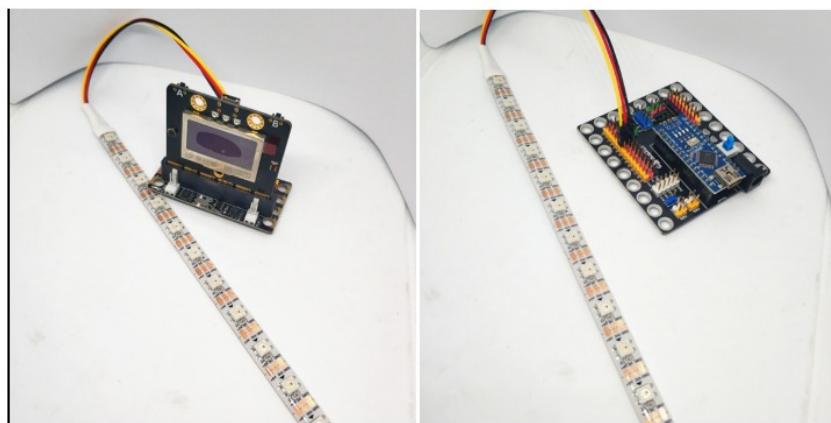


## 7. WS2812-RGB灯带



WS2812灯带由12颗级联的LED单元组成，每颗LED灯为5mmx5mm的正方形，每颗灯能单独发出红色(R),绿色(G),蓝色(B)，而且每颗灯都集成了WS2812控制芯片，12颗灯级联后可通过一根数据线连接开发板，通过程序来控制灯带的亮起效果。

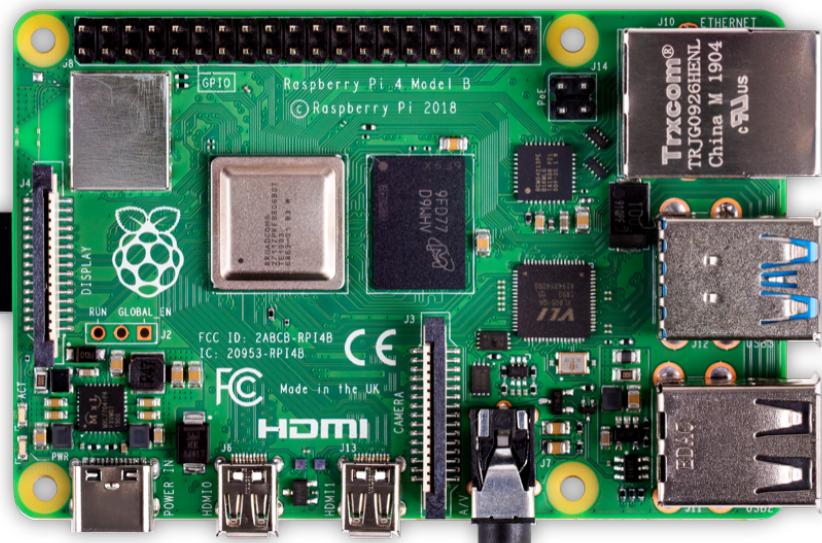
WS2812灯带的接口定义如下表所示： |线的颜色|定义| :-:|:-: |黄色|数据线| |红色|电  
源5V| |黑色|电源负极|



© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook修订时  
间： 2020-03-01

## 第5节 认识硬件—树莓派（Raspberry Pi）

### Raspberry Pi 4（树莓派4代）

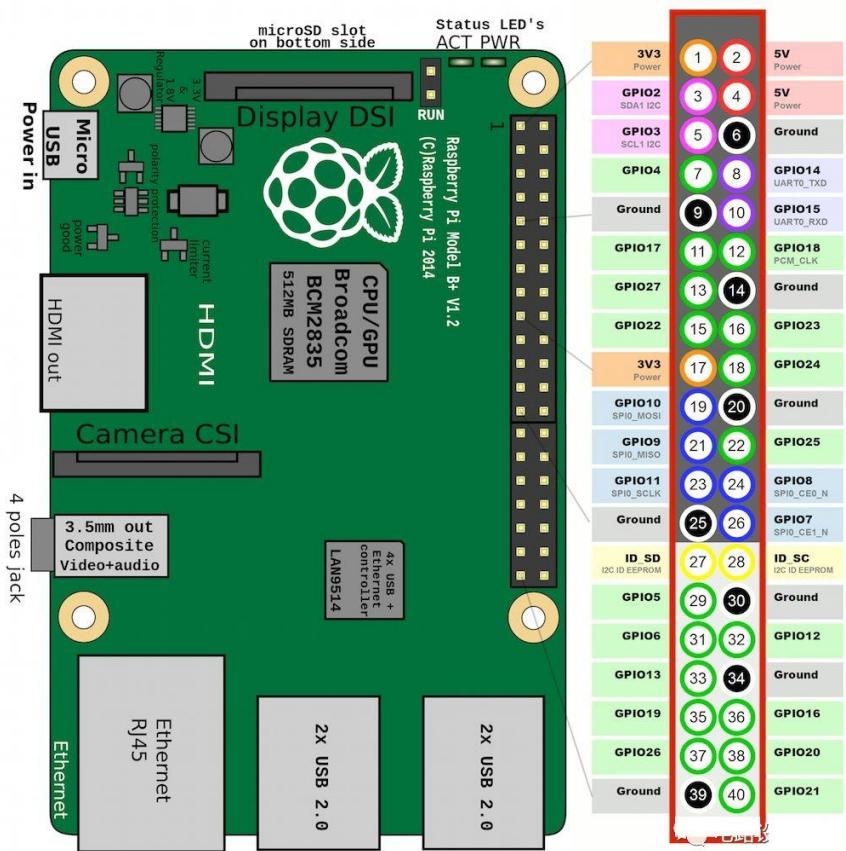


Raspberry Pi（树莓派）是早在2012年，由树莓派基金会发售的一台只有信用卡大小的电脑，他们发售树莓派的目的是给贫困地区的儿童一台廉价的用于学习编程的电脑，经过几年的迭代，树莓派已经到了第四代。（如上图所示）

树莓派的详细硬件配置见于下表：

项目	参数	名称	Raspberry Pi4 (树莓派4)	工作电压	3.3V 和 5v	输入电压	5V	GPIO数量	40 Pin	支持的接口	I2C, SPI, UART	主频	4核-1.4GHZ	运行内存	4GB	数据存储	64GB-TF卡	外部接口	USBx4, 网口, mini-HDMIx2, 3.5mm音视频接口	外部供电接口	Type-C 电压： 5V	供电方式	10000毫安移动电源	操作系统	Raspbian (基于Debian定制的Linux操作系统)	操作方式	接入显示器和键盘鼠标；局域网内使用VNC和远程桌面访问
----	----	----	----------------------	------	-----------	------	----	--------	--------	-------	----------------	----	-----------	------	-----	------	----------	------	------------------------------------	--------	---------------	------	-------------	------	---------------------------------	------	-----------------------------

#### 树莓派GPIO接口



GPIO (General-purpose input/output)，即通用输入输出接口。

树莓派的GPIO名称与功能如下表所示：

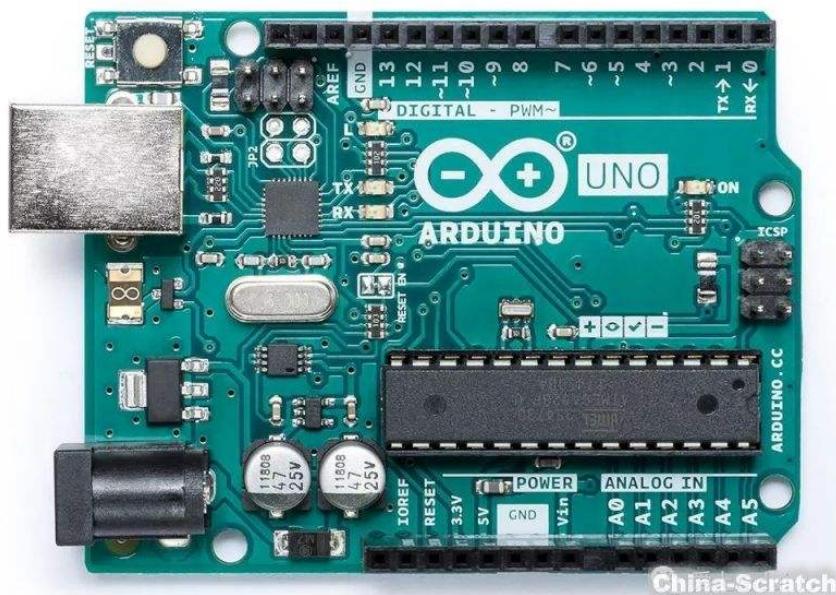
GPIO	功能	
	3.3V	3.3V供电 (可为3.3V工作电压的设备提供电源, 即电源正极)
	5V	5V供电 (可为5V工作电压的设备提供电源, 即电源正极)
	Ground(GND)	电源负极, 且所有Ground相通, 接在任意Ground皆可
	注意	3.3V, 5V与Ground引脚不可短接, 否则会烧毁树莓派主板
	GPIO2~GPIO27	可用于编程控制的GPIO口
	输入电压	5V
	GPIO数量	40 Pin
	支持的接口	I2C, SPI, UART
	主频	4核-1.4GHZ
	运行内存	4GB
	数据存储	64GB-TF卡
	外部接口	USBx4, 网口, mini-HDMIx2, 3.5mm音视频接口
	外部供电接口	Type-C 电压: 5V
	供电方式	10000毫安移动电源
	操作系统	Raspbian (基于Debian定制的Linux操作系统)
	操作方式	接入显示器和键盘鼠标; 局域网内使用VNC和远程桌面访问

注: 树莓派系统的基本操作详见于本章“第二节 Linux基本操作”

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook修订时间: 2020-03-13

## 第6节 认识硬件—Arduino

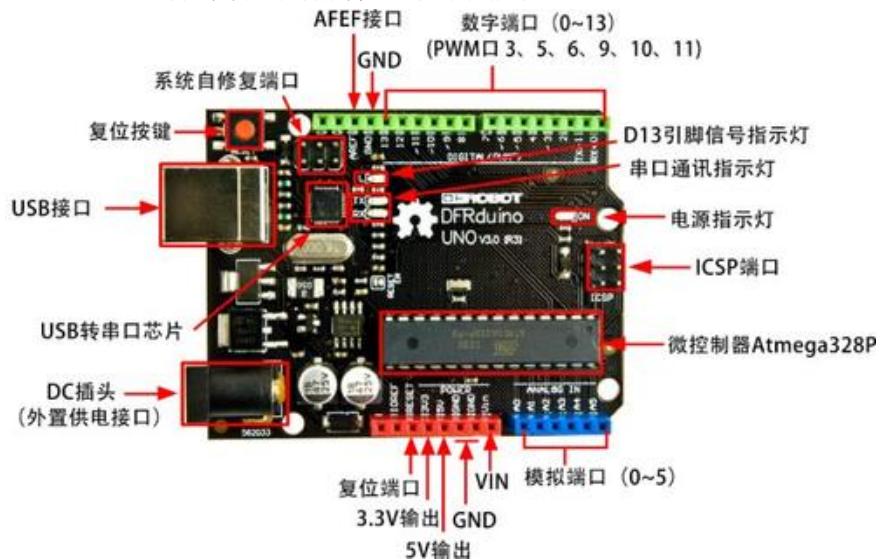
### 1. Arduino UNO



Arduino UNO 上搭载一块Atmel公司生产的AVR单片机，名称为ATmega328P。详细信息如下表所示

项目	参数
名称	Arduino UNO (ATmega328P)
工作电压	5V
输入电压	7V~12V
数字I/O引脚数量	14 Pin
PWM通道	6 Pin
模拟I/O引脚数量	6 Pin
所有I/O口电流输出大小	20 mA
Flash大小	32 KB
SRAM	2 KB
EEPROM	1 KB
时钟速度	16 MHz
板载LED灯控制引脚	13号数字I/O口
程序下载接口	USB 标准B型口
外部供电接口	5.2mm DC接口
程序编码与编译环境	Arduino IDE (IDE:集成开发环境)

Arduino UNO 的各个接口和功能介绍, 如下图所示:

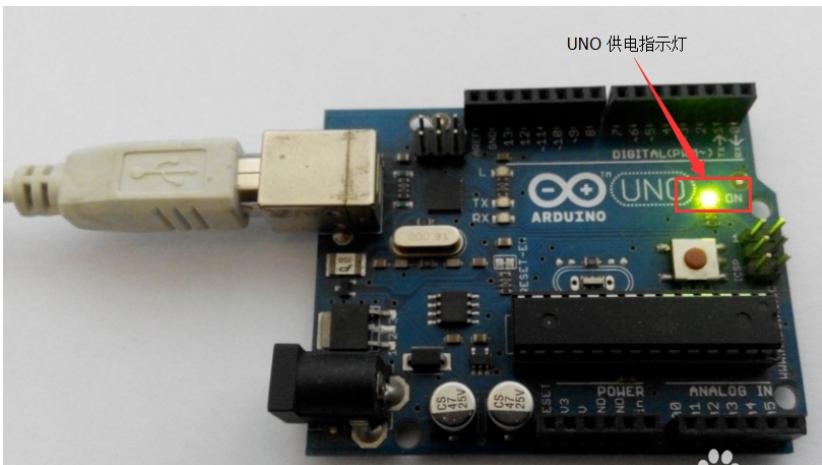


## 程序烧录过程

- 在桌面上打开“Arduino IDE”(此集成开发环境已提前配置完整, 直接使用即可), 如下图所示:



2. 将Arduino UNO 通过USB线连接到电脑的USB口上，如下图：



3. 通过Arduino IDE打开程序代码，如使13号接口连接的LED灯闪烁的程序：

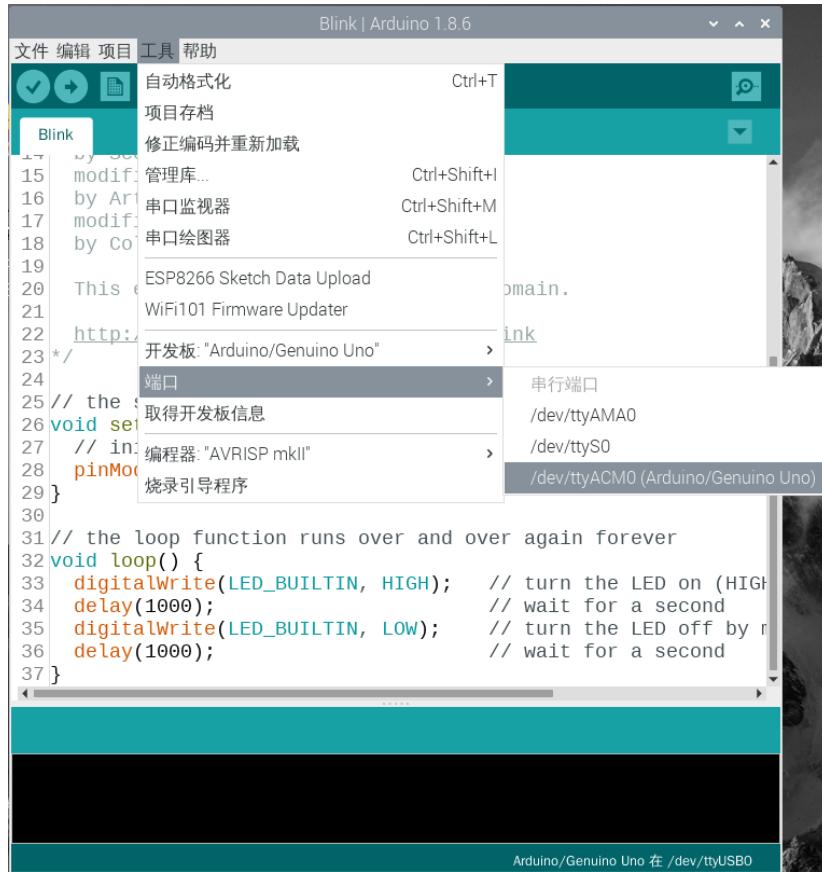
Blink



4. 选择目标开发板：Arduino UNO,如下图：



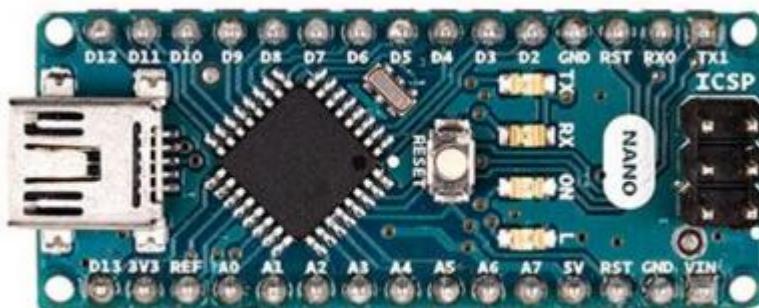
5. 选择开发板所在的物理端口，如下图：



6. 点击下载按钮，Arduino IDE 开始编译源程序并把编译结果下载到开发板上；  
完成后，开发板上13号I/O口连接的LED会每隔一秒亮灭一次，如下图：



## 2. Arduino Nano

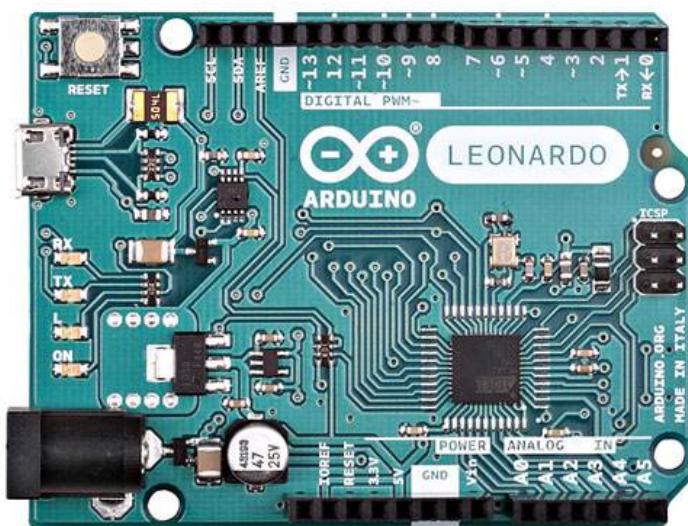


Arduino Nano 与 Arduino UNO 同样搭载一块Atmel公司生产的AVR单片机，名称为ATmega328P。只是与Arduino UNO 相比，Arduino Nano在同样搭载了USB程序下载接口，更多一些的I/O口的前提下，体积要小的多，这就为需要小体积控制板的项目提供了更多选择。详细信息如下表所示

项目	参数
名称	Arduino UNO (ATmega328P)
工作电压	5V
输入电压	7V~12V
特殊接口	原生支持USB接口
数字I/O引脚数量	22 Pin
PWM通道	6 Pin
模拟I/O引脚数量	6 Pin
所有I/O口电流输出大小	40 mA
Flash大小	32 KB
SRAM	2 KB
EEPROM	1 KB
时钟速度	16 MHz
板载LED灯控制引脚	13号数字I/O口
程序下载接口	Micro-USB接口
外部供电接口	5.2mm DC接口
程序编码与编译环境	Arduino IDE (IDE:集成开发环境)

注：Arduino Nano的程序烧写方法与Arduino UNO 相同。

### 3. Arduino Leonardo



Arduino Leonardo 是一个搭载ATmega32u4的8位AVR单片机的开发板，其上同样有与Arduino UNO 相同的I/O口，唯一不同的是，ATmega32u4单片机支持原生的

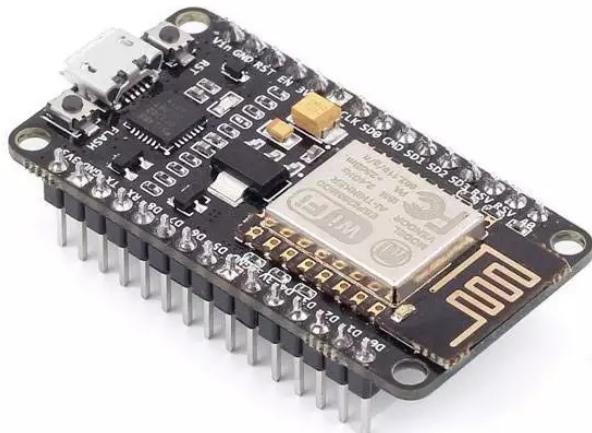
**USB**接口，并且可以通过程序控制，来讲此**USB**接口模拟成各类电脑**USB**外设，如鼠标，键盘，游戏手柄等等。详细信息如下图表格所示： |项目|参数|:-:|-:|名  
称|Arduino Leonardo (ATmega32u4)| |工作电压|5V| |输入电压|7V~12V| |数字I/O引  
脚数量|14 Pin | |PWM通道|6 Pin| |模拟I/O引脚数量|8 Pin| |所有I/O口电流输出大  
小|40 mA| |Flash大小|32 KB| |SRAM|2 KB| |EEPROM|1 KB| |时钟速度|16 MHz| |  
板载LED灯控制引脚|13号数字I/O口|| |程序下载接口|Mini-USB接口|| |外部供电接  
口|5.2mm DC接口|| |程序编码与编译环境|Arduino IDE (IDE:集成开发环境)|

注：Arduino Leonardo的程序烧写方法与Arduino UNO 相同

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook修订时  
间： 2020-03-01

## 第7节 认识硬件—ESP8266

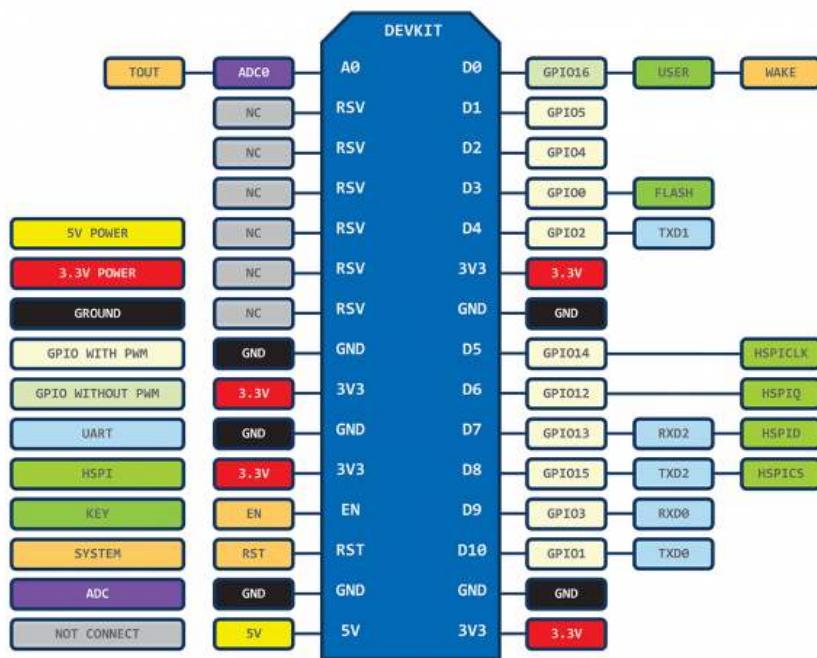
### 1. NodeMCU-Esp8266开发板



NodeMcu开发板上搭载了一颗国产32位WiFi SOC(片上系统): ESP8266; 也附带了TTL串口转USB芯片，可以直接通过Micro-USB线(安卓数据线)连接电脑，通过PC端的软件为其编译下载程序。由于ESP8266原生支持wifi,所以其在全球物联网领域大放异彩。详细参数如下表所示：

项目	参数
名称	NodeMcu(ESP8266)
工作电压	3.3V
输入电压	5V
数字I/O引脚数量	11 Pin
模拟I/O引脚数量	1 Pin
支持的接口	I2C, SPI, UART
Flash大小	4MB
时钟速度	120 MHz
程序下载接口	Micro-USB接口
外部供电接口	VIN插针 电压为5V
程序编码与编译环境	Arduino IDE (IDE:集成开发环境)

NodeMcu的Arduino引脚编号与芯片引脚编号的对应关系如下图所示：



## 给NodeMcu烧写程序

在本课程中我们使用Arduino IDE为NodeMcu烧写程序。

1. 在桌面上打开“Arduino IDE”(此集成开发环境已提前配置完整，直接使用即可)，  
如下图所示：



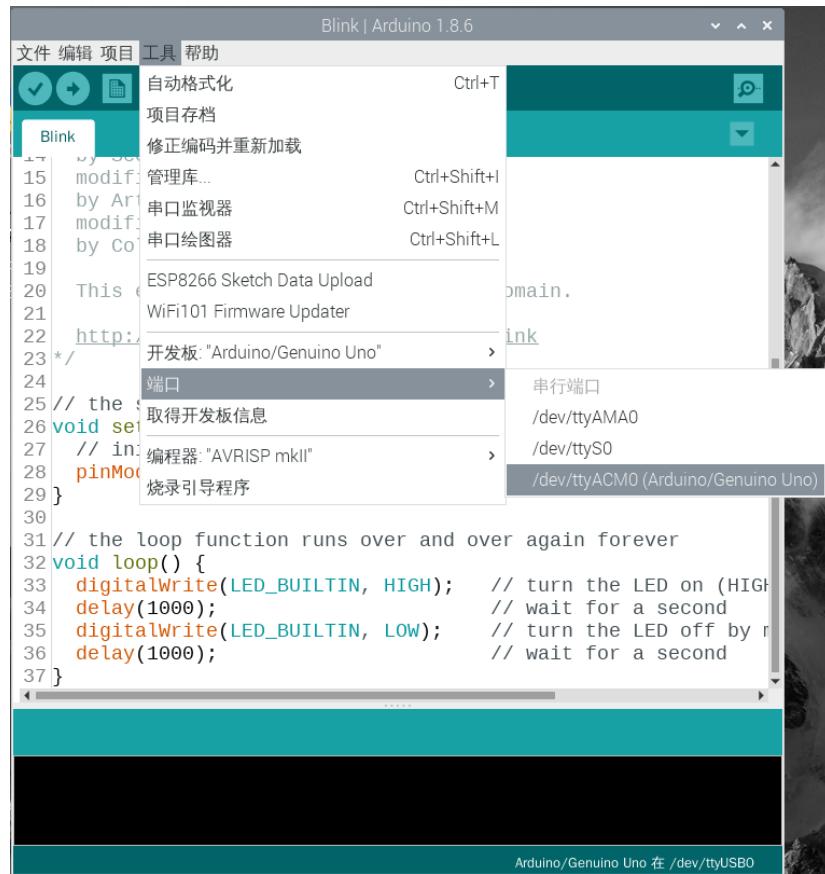
2. 将NodeMcu 通过Micro-USB线连接到电脑的USB口上，用Arduino IDE打开任意扩展名为“\*.ino”程序。



1. 选择目标开发板: "NodeMcu 1.0(ESP-12e Module)",如下图:



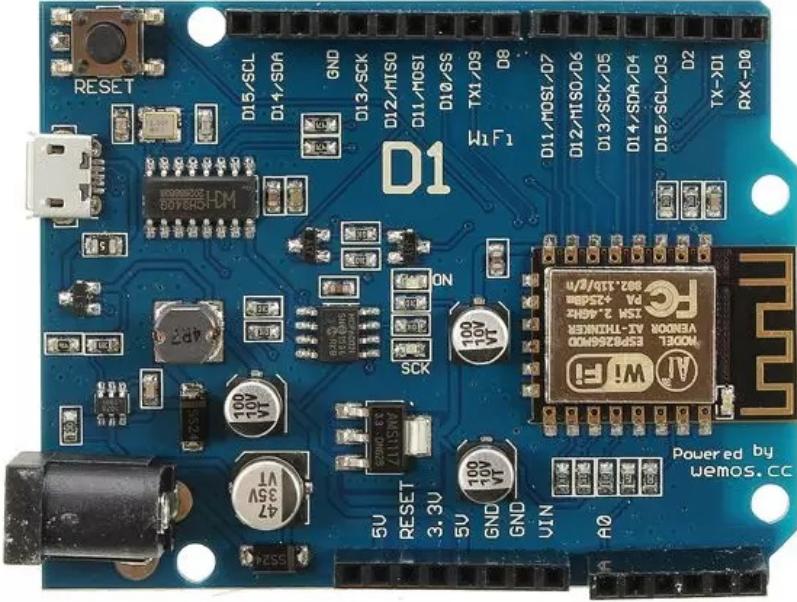
2. 选择开发板所在的物理端口"/dev/ttyUSB0", 如下图:



3. 点击下载按钮，Arduino IDE 开始编译源程序并把编译结果下载到开发板上。



## 2. Wemos D1-Esp8266开发板



Wemos D1与NodeMcu开发板具有相同的主控核心：Esp8266wifi模块，所以NodeMcu能实现的功能，Wemos D1也同样能实现，不同的是Wemos D1与Arduino UNO有相同的板型和接口，所以Arduino UNO上能用的扩展板或模块，Wemos D1也同样适用。Wemos D1的详细参数如下表所示：

项目	参数
主控核心	ESP8266
数字I/O数量	13 Pin
模拟I/O数量	1 Pin
复位按键(reset)	位于板左上角的微动开关(按一次主控重启，程序从头运行)
程序下载和低电流供电接口	Micro-USB 接口
外部供电接口	5.1mm DC接口
特殊功能	板载WiFi模块

## Wemos D1烧写程序

Wwemos D1的程序烧写过程与NodeMcu大致相同，除要选择的板卡不同外，其他都相同。

板卡选择如下图所示：

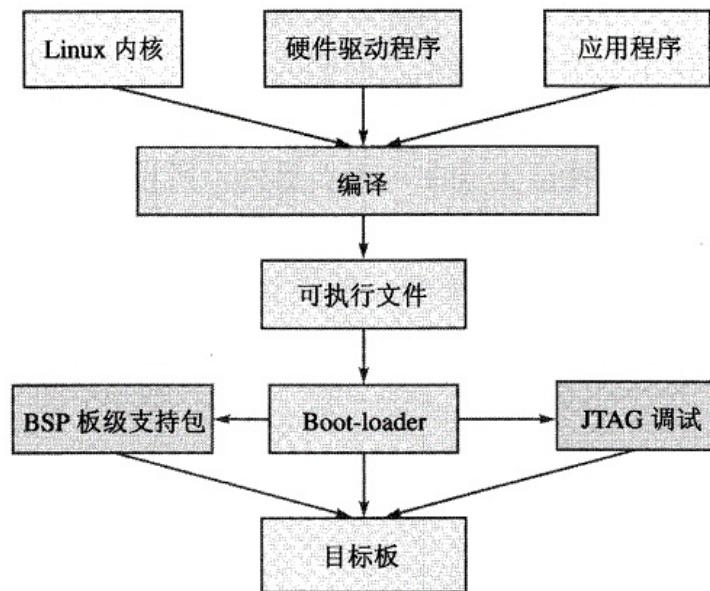


© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook 修订时间： 2020-03-01

## 第8节 嵌入式系统软件开发流程

嵌入式系统定义：嵌入式系统（**Embedded Systems**）由硬件和软件组成，是能够独立进行运作的器件。其软件内容只包括软件运行环境及其操作系统。硬件内容包括信号处理器、存储器、通信模块等在内的多方面的内容。

嵌入式系统软件开发流程框图：



## 嵌入式硬件平台

按照微处理器指令集类型来划分，分为复杂指令集（**CISC**，即Complex Instruction Set Computing）和精简指令集（**RISC**即Reduced Instruction Set Computing）。复杂指令集的处理器架构以x86和AMD64架构为主，主要应用在个人电脑，网站服务器等场景，复杂指令集的处理器的每条指令按顺序串行执行，控制相对简单，但速度相对较慢。

精简指令集的处理器架构有ARM, AVR, MIPS等，这些架构各有特点；而在本次

课程中用到的树莓派4开发板的处理器就属于ARM架构。本节主要以ARM架构为例讲解嵌入式系统软件开发流程。



## 嵌入式软件

嵌入式软件：嵌入式软件就是在嵌入式硬件中运行的操作系统和在嵌入式操作系统中的应用程序、驱动程序和应用程序等。

## 嵌入式操作系统

到目前嵌入式领域中的嵌入式操作系统有：Linux, Unix, Windows Embedded, VXWorks, RTOS等。而占据主要地位的是开源阵营的以Linux为核心的操作系统，如Android, Openwrt, Ubuntu, Debian, RedHat, CentOS等等。

本次课程中用到的树莓派所用的操作系统就是以Debian为主深度优化的嵌入式操作系统：Raspbian，其系统核心仍然是Linux。

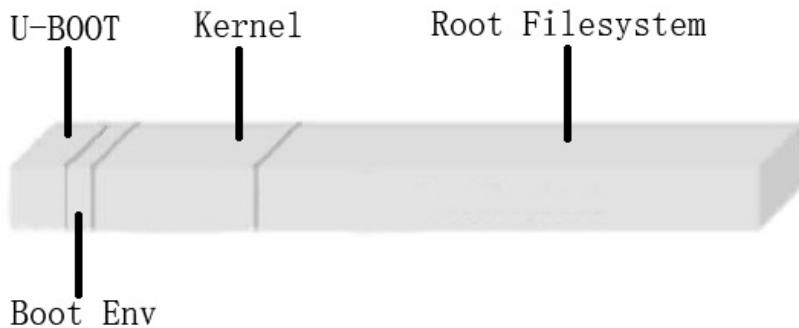
## 操作系统是如何被制作出来的

### 嵌入式ARM-Linux操作系统的结构

ARM-Linux操作系统由一下几部分构成：

- |名称|说明| :-:|-:|
- |Bootloader|引导加载器：这是一段裸机程序，能够在硬件上电后完成对硬件的检测，初始化各个硬件模块，主要功能是完成加载并引导操作系统启动的工作；常见的Botloader有U-boot, Bread, Grub, BIOS等|
- |Kernel|linux操作系统的内核：包含操作系统的各项功能，硬件的驱动等|
- |Rootfs|根文件系统：根文件系统提供对文件的管理、对文件系统的支持，以及提供Linux系统下的标准目录结构(详见本章第二节：Linux基本操作)|

下图为嵌入式linux操作系统在存储设备上的分区结构图：



注：Boot Env分区为U-boot环境变量存储分区，引导操作系统启动所需的相关配置信息就存储在此分区内。

## 从源代码开始

由于Linux是开源操作系统，其源代码可以按照开源协议任意修改，也可以用于我们的嵌入式操作系统。

操作系统的源码大多以C语言这种高级语言编写，所以从操作系统源代码开始，对源代码进行一定的“配置”，再通过编译器的“预处理”，“编译”，“汇编”，“链接”四步的深度加工，最后再经过固件打包工具的处理才能制作出能用的嵌入式操作系统。

由于嵌入式处理器大多为精简指令集(RISC)处理器，其处理能力相对PC端处理器较弱，不适合用来编译操作系统固件，所以我们要在X86平台上编译ARM架构处理器能执行的操作系统软件，这种编译方式称为“交叉编译”。

## 交叉编译的工具—交叉编译工具链

要把源代码制作成嵌入式操作系统的可烧写镜像(.img文件)，中间就需要交叉编译工具链(Toolchains)的操作，常用免费的的交叉编译工具链有：|编译器|作用|:-|:  
|arm-none-linux-gnueabi-gcc|1. Codesourcery 公司基于GCC推出的的ARM交叉编译工具。

2.用于交叉编译ARM(32位)系统中所有环节的代码，包括裸机程序、u-boot、Linux kernel、filesystem和App应用程序| |arm-linux-gnueabihf-gcc|1.Linaro公司基于GCC推出的的ARM交叉编译工具。

2.用于交叉编译ARM(32位)系统中所有环节的代码，包括裸机程序、u-boot、Linux kernel、filesystem和App应用程序| |aarch64-linux-gnu-gcc|1.Linaro公司基于GCC推出的的ARM交叉编译工具。

2.用于交叉编译ARMv8 64位目标中的裸机程序、u-boot、Linux kernel、filesystem和App应用程序| |arm-none-elf-gcc|1.Codesourcery公司基于GCC推出的的ARM交叉编译工具。

2.用于交叉编译ARM MCU(32位)芯片，如ARM7、ARM9、Cortex-M/R芯片程序| |arm-none-eabi-gcc|1.GNU推出的的ARM交叉编译工具。

2.用于交叉编译ARM MCU(32位)芯片，如ARM7、ARM9、Cortex-M/R芯片程序|

由于经交叉编译器处理U-boot、内核和根文件系统源码后只生成的其二进制文件，因此还需要将U-boot、内核(Kernel)和根文件系统(Rootfs)三者打包整合，最终生成可用的嵌入式操作系统固件(.img文件)。

## 嵌入式操作系统调试

### TFTP和NFS远程挂载根文件系统

由于嵌入式开发板的存储设备容量较小，若交叉编译生成的系统镜像较大，每次将系统镜像烧写到板载存储设备中比较耗时，调试起来效率较低；因此，为避免这些问题，常使用TFTP和NFS远程挂载根文件系统的方式进行开发调试。步骤详见下表：

|名称|配置步骤| :-:- |Linux PC端|1. 配置并开启TFTP和NFS服务；

2. 设置有线网卡IP地址，使其与开发板网卡IP在同一网段；

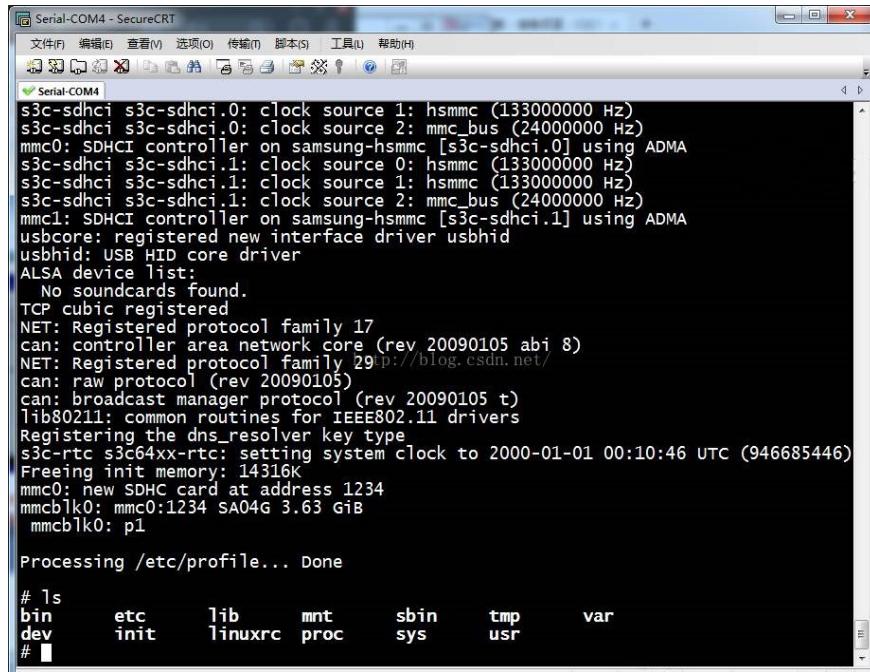
3. 将编译好的内核镜像(Kernel.img)放入TFTP服务指定的目录下；

4. 将开发板可用的根文件系统放置于PC端的某个目录下||开发板|1. 开发板启动到支持网络的U-boot下；

2. 配置U-boot Env，使其IP地址与PC网卡IP在同一网段；

2. 配置U-boot Env,从tftp服务器获取并启动内核和从NFS服务器获取并挂载根文件系统；

3. 在U-boot的命令行中键入启动命令，启动内核和挂载根文件系统。|



The screenshot shows a terminal window titled "Serial-COM - SecureCRT". The window contains the following text:

```
s3c-sdhci s3c-sdhci.0: clock source 1: hsmmc (133000000 Hz)
s3c-sdhci s3c-sdhci.0: clock source 2: mmc_bus (24000000 Hz)
mmc0: SDHCI controller on samsung-hsmmc [s3c-sdhci.0] using ADMA
s3c-sdhci s3c-sdhci.1: clock source 0: hsmmc (133000000 Hz)
s3c-sdhci s3c-sdhci.1: clock source 1: hsmmc (133000000 Hz)
s3c-sdhci s3c-sdhci.1: clock source 2: mmc_bus (24000000 Hz)
mmc1: SDHCI controller on samsung-hsmmc [s3c-sdhci.1] using ADMA
usbcore: registered new interface driver ushbd
usbhid: USB HID core driver
ALSA device list:
  No soundcards found.
TCP cubic registered
NET: Registered protocol family 17
can: controller area network core (rev 20090105 abi 8)
NET: Registered protocol family 29 tp://blog.csdn.net/
can: raw protocol (rev 20090105)
can: broadcast manager protocol (rev 20090105 t)
lib80211: common routines for IEEE802.11 drivers
Registering the dns_resolver key type
s3c-rtc s3c64xx-rtc: setting system clock to 2000-01-01 00:10:46 UTC (946685446)
Freeing init memory: 14316K
mmc0: new SDHC card at address 1234
mmcblk0: mmc0:1234 SA04G 3.63 GiB
  mmcblk0: p1

Processing /etc/profile... Done

# ls
bin      etc      lib      mnt      sbin      tmp      var
dev      init     linuxrc  proc     sys       usr
#
```

上图为通过NFS方式远程挂载根文件系统(Rootfs)并启动内核。

## 嵌入式Linux操作系统基本操作

### 通过SSH方式登录Linux系统

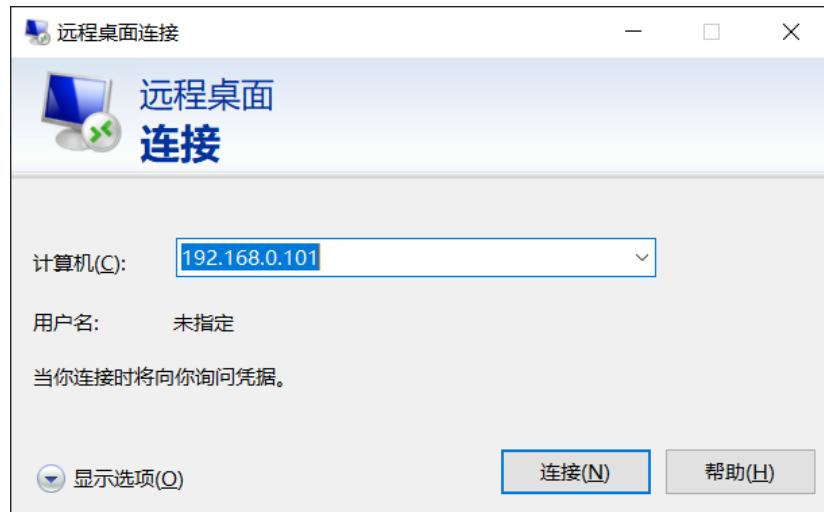
1. SSH：为安全外壳协议(Secure Shell)；由 IETF 的网络小组（Network Working Group）所制定；SSH 为建立在应用层基础上的安全协议。SSH 是较可靠，专为远程登录会话和其他网络服务提供安全性的协议。利用 SSH 协议可以有

效防止远程管理过程中的信息泄露问题。

下图为使用Putty通过SSH方式访问树莓派：



2. xrdp:是Linux平台上的远程桌面的一种，通过在系统中开启此软件(服务)，就可  
以通过Windows操作系统中的“远程桌面”应用程序访问目标计算机的桌面，如  
下图所示：



如上图所示，在局域网中，只需在“计算机”文本框中填入目标计算机的IP地址，即  
可访问到远程计算机的桌面。

若远程桌面访问起来比较卡顿，可通过修改“显示选项”来改善：

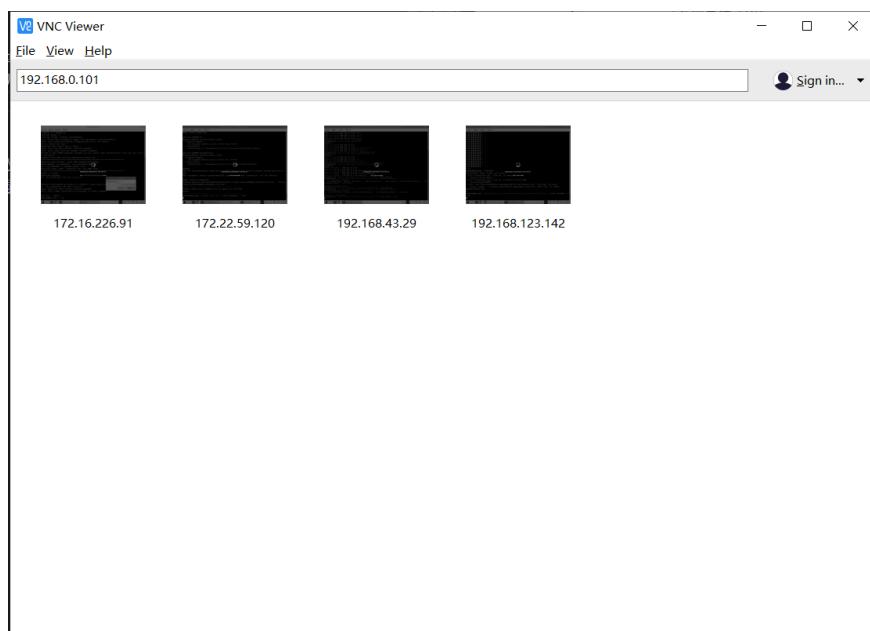
### 1. 修改显示分辨率

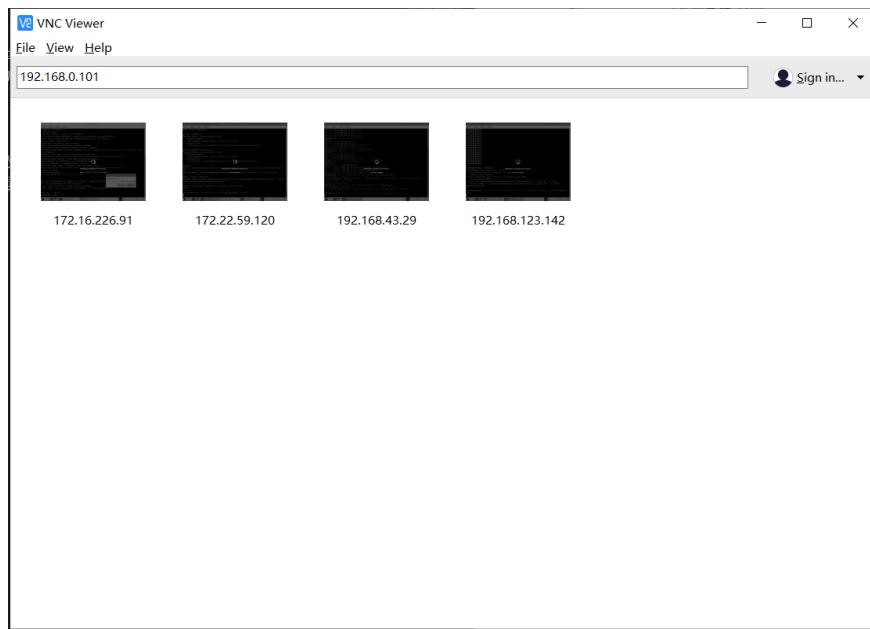


## 2. 修改网络传输类型



3.VNC：是第三方公司开发的远程桌面访问软件，树莓派官方系统直接对VNC服务端做了系统整合；只需在局域网的PC端安装“VNC Viewer”，即可通过此PC访问目标计算机在局域网中的IP地址来开启远程桌面。如下图所示：





© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-03-13

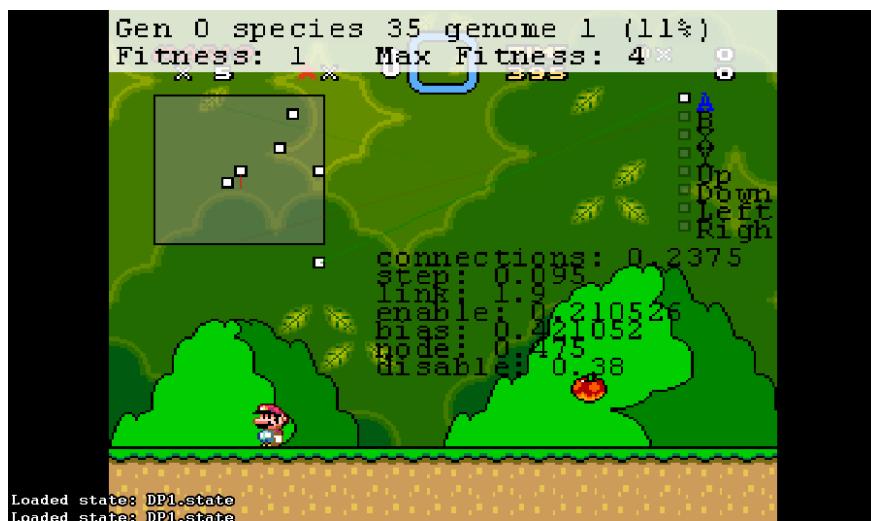
# 第2章 人工智能初体验

## 本章主题：体验

学习者通过学习本章内容，将进一步了解人工智能和机器学习的发展历程、相关概念以及主要应用。此外，以样例体验的形式进行学习，学习者将对人工智能在图形图像和语言处理、电子游戏及其他领域的应用产生更加深刻的认识与理解。

## 本章重点

- 1 了解人工智能的基本概念和原理
- 2 了解卷积神经网络（CNN）以及深度神经网络（RNN）等的应用



## 涉及软硬件

- PC
- 树莓派（Raspberry Pi）
- NVIDIA Jetson Nano

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2020-02-26

# 第1节 人工智能、机器学习的相关概念

机器学习是人工智能研究发展到一定阶段的必然产物。

二十世纪五十年代到七十年代初，人工智能研究处于推理期，那时人们认为只要能赋予机器逻辑推理能力，机器就具有智能。

随着研究发展，在七十年代中期开始，人工智能研究进入了知识期，要使机器具有智能，就必须设法使机器拥有知识。此期间大量的专家系统面世。

在八十年代，从样例中学习（监督和无监督学习等）的一大主流是符号主义学习。其代表包括决策树(**decision tree**)和基于逻辑的学习。

九十年代中期之前，从样例中学习的另一主流技术是基于神经网络的连接主义学习。与符号主义学习能产生明确的概念表示不同，连接主义学习产生的是黑箱模型。连接主义最大的局限是试错性：学习过程涉及大量参数，而参数的设置缺乏理论指导，主要靠手工调试。

九十年代中期，统计学习(**statistical learning**)登场。代表技术是支持向量机(**support vector machine**)。

二十一世纪初，连接主义通过深度学习卷土重来。所谓深度学习，即很多层的神经网络。在涉及语音、图像等复杂对象的应用中，深度学习取得了优越性能。深度学习虽然缺乏严格的理论基础，但是它显著降低了机器学习的门槛，为机器学习的实践带来了便利。

当前时代，互联网和硬件高度发达，人们进入了大数据时代，深度学习取得了大发展。随着物联网、边缘计算、5G网络、IPV6等的发展和普及，相信人工智能会在人类社会发挥更大的作用。

## 算法

机器学习致力于研究如何通过计算的手段，利用经验来改善系统自身的性能。在计算机系统中，经验通常以数据形式存在。因此，机器学习所研究的主要内容，是关于在计算机上从数据中产生模型(**model**)的算法。有了算法，我们把经验数据提供给它，它就能基于这些数据产生模型；在面对新的情况时，模型会给我们提供对应的判断。

## 数据集

要进行机器学习，先要有数据。假定我们收集了一批关于西瓜的数据，例如

```
{色泽=青绿; 根蒂=蜷缩; 敲声=浊响}  
{色泽=乌黑; 根蒂=稍蜷; 敲声=沉闷}  
{色泽=浅白; 根蒂=硬挺; 敲声=清脆}
```

这样的一组数据称为一个数据集(**data set**)，其中每条记录是关于一个事件或对象的描述，称为一个示例(**instance**)或样本(**sample**)。如果把每个样本中的色泽、根蒂和敲声作为三个坐标轴，则它们张成一个用于描述西瓜的三维空间，每个西瓜都

可以在这个空间中找到自己的位置。当然，一般来说，维数越多，描述就会越精确。空间中每个点对应一个坐标向量，因此我们也把一个样本称为特征向量  
**(feature vector)**

## 训练

从数据中学得模型的过程称为学习(**learning**)或训练(**training**)，这个过程通过执行某个算法来完成。训练过程中使用的数据称为训练数据(**training data**)，其中每个样本称为一个训练样本(**training set**)。学得的模型会对应关于数据的某种规律。

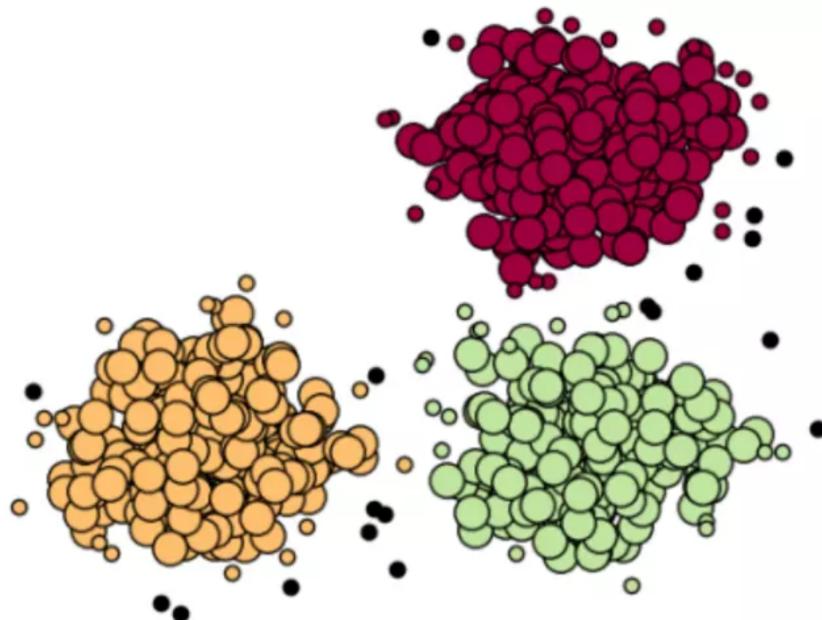
例如，如果希望学得一个能帮助我们判断一个西瓜是不是好瓜的模型，仅有前面的数据集是不够的。要建立关于预测(**prediction**)的模型，我们需要过得训练样本的结果信息：例如 {{色泽=青绿；根蒂=蜷缩；敲声=浊响}，好瓜} 这个关于结果的信息（好瓜）称为标记(**label**)。

## 分类、回归、聚类、监督与无监督学习

若我们预测的是离散值，例如好瓜、坏瓜，此类学习任务称为分类  
**(classification)**

若预测的是连续值，如西瓜成熟度0.95, 0.37，此类学习任务称为回归  
**(regression)**

我们可以对西瓜做聚类(**clustering**)。即将训练集中的西瓜分为若干组，每组称为一个簇(**cluster**)。例如，算法自动将数据集分成了3簇，用三种颜色代表。每一簇内较大的点代表核心对象，较小的点代表边界点。黑色的点代表离群点或者叫噪声点。



根据训练数据是否拥有标记信息（好瓜），学习任务可划分为两大类：监督学习  
**(supervised learning)**和无监督学习(**unsupervised learning**)

分类和回归是监督学习的代表。聚类是无监督学习的代表。

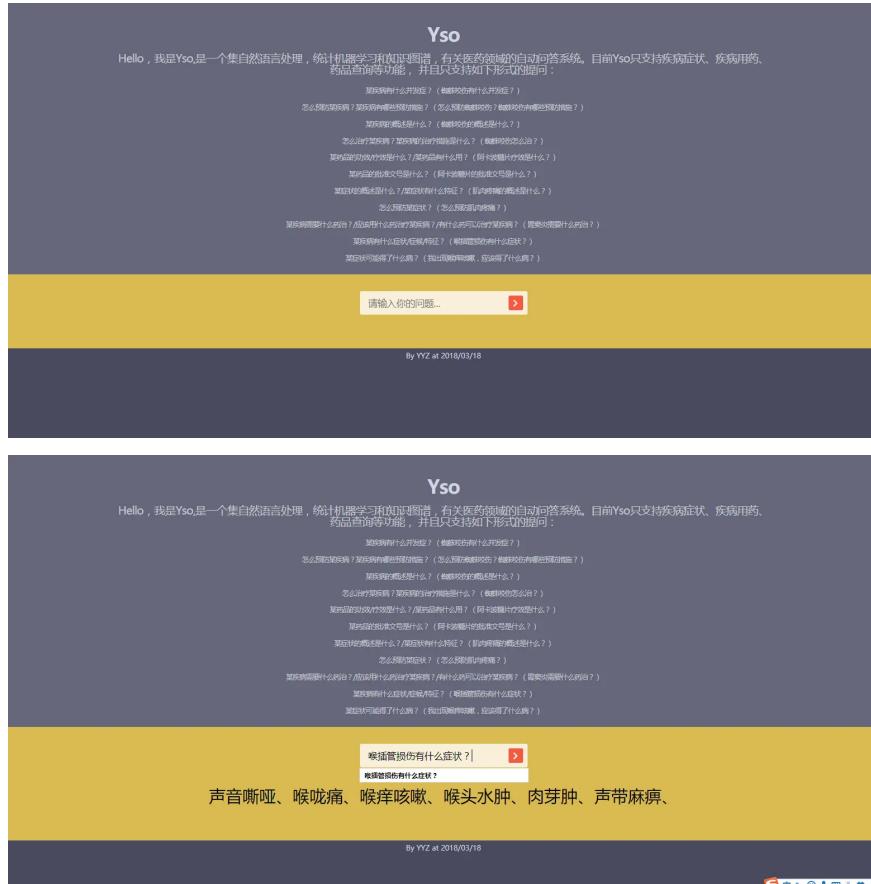
© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook 修订时间： 2020-02-26

## 第2节 知识图谱

知识图谱是由 Google 公司在 2012 年提出来的一个新的概念。从学术的角度，我们可以对知识图谱给一个这样的定义：知识图谱本质上是语义网络（**Semantic Network**）的知识库。

### 知识图谱在线体验

基于医药知识图谱的智能问答系统

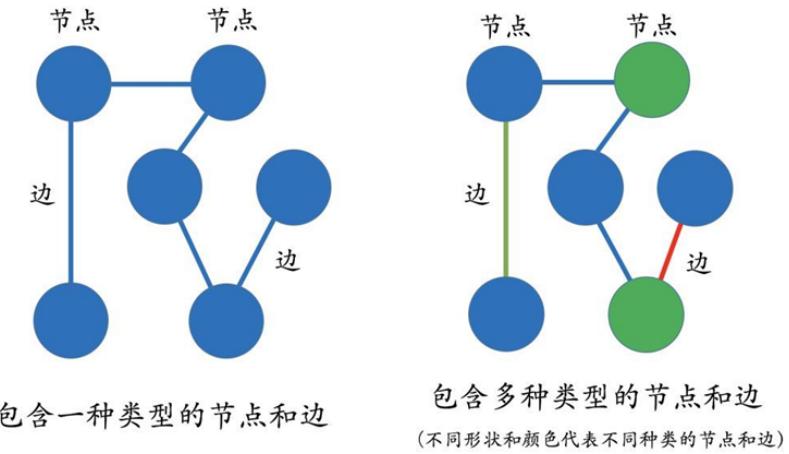


这是一个集自然语言处理，统计机器学习和知识图谱，有关医药领域的初级自动问答系统。

该问答系统可以解析输入的自然语言问句生成相应的后台查询指令，进一步请求后台基于**TDB**知识库相关服务，进而得到问题的结果。

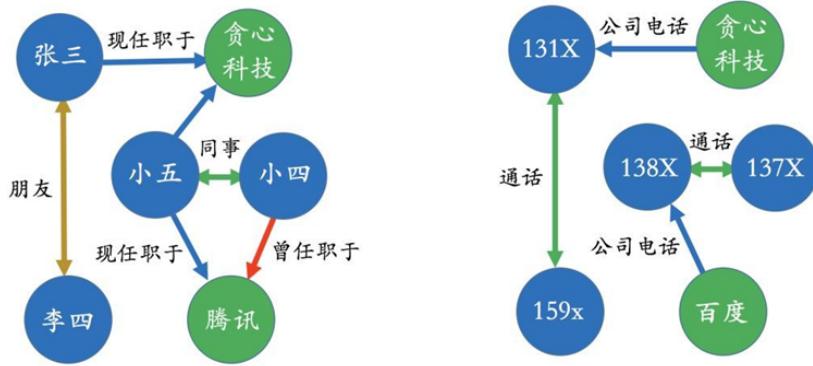
从实际应用的角度出发其实可以简单地把知识图谱理解成多关系图（**Multi-relational Graph**）。那什么叫多关系图呢？学图是由节点（**Vertex**）和边（**Edge**）来构成，但这些图通常只包含一种类型的节点和边。但相反，多关系图

一般包含多种类型的节点和多种类型的边。比如左下图表示一个经典的图结构，右边的图则表示多关系图，因为图里包含了多种类型的节点和边。这些类型由不同的颜色来标记。



在知识图谱里，我们通常用“实体（Entity）”来表达图里的节点、用“关系（Relation）”来表达图里的“边”。实体指的是现实世界中的事物比如人、地名、概念、药物、公司等，关系则用来表达不同实体之间的某种联系，比如人 -“居住在”- 北京、张三和李四是“朋友”、逻辑回归是深度学习的“先导知识”等等。

现实世界中的很多场景非常适合用知识图谱来表达。比如一个社交网络图谱里，我们既可以有“人”的实体，也可以包含“公司”实体。人和人之间的关系可以是“朋友”，也可以是“同事”关系。人和公司之间的关系可以是“现任职于”或者“曾任职”的关系。类似的，一个风控知识图谱可以包含“电话”、“公司”的实体，电话和电话之间的关系可以是“通话”关系，而且每个公司它也会有固定的电话。



## 我们为什么需要知识图谱

当你看到下面这行文本时会想到什么？

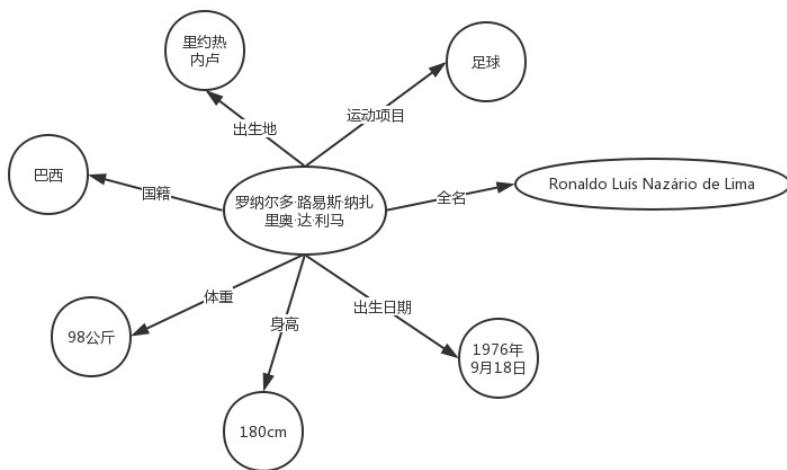
Ronaldo Luís Nazário de Lima

估计绝大多数中国人不明白上面的文本代表什么意思。没关系，我们看看它对应的中文：

## 罗纳尔多·路易斯·纳萨里奥·德·利马

现在大多数人应该能够明白这是一个外国人的名字。熟悉足球的人可能会知道这是一个巴西足球运动员。

之所以举这样一个例子，是因为，计算机一直面临着这样的困境——无法获取网络文本的语义信息。尽管近些年人工智能得到了长足的发展，在某些任务上取得超越人类的成绩，但离一台机器拥有一个两三岁小孩的智力这样一个目标还有一段距离。这距离的背后很大一部分原因是机器缺少知识。如同上面的例子，机器看到文本的反应和我们看到罗纳尔多葡萄牙语原名的反应别无二致。为了让机器能够理解文本背后的含义，我们需要对可描述的事物（实体）进行建模，填充它的属性，拓展它和其他事物的联系，即，构建机器的先验知识。就以罗纳尔多这个例子说明，当我们围绕这个实体进行相应的扩展，我们就可以得到下面这张知识图。

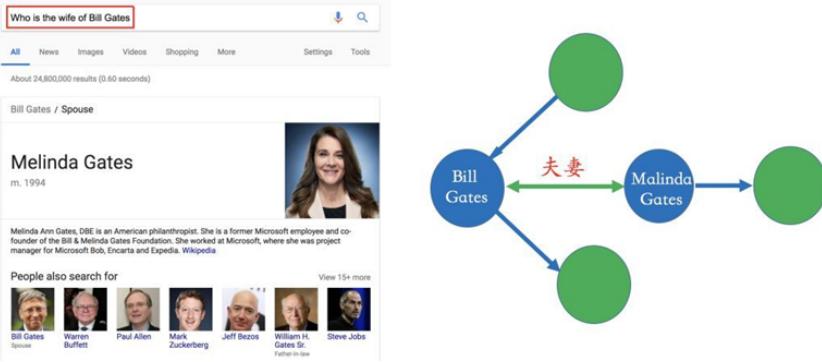


机器拥有了这样的先验知识，当它再次看到 Ronaldo Luís Nazário de Lima，它就会“想”：“这是一个名字叫 Ronaldo Luís Nazário de Lima 的巴西足球运动员。”这和我们人类在看到熟悉的事物，会做一些联想和推理是很类似的。

需要说明的是，上面的知识图这是一张草图，并不代表知识图谱的实际组织形式，相反，它还会让读者对知识图谱产生一定的误解。

Google 为了提升搜索引擎返回的答案质量，推出了知识图谱概念。有知识图谱的辅助，搜索引擎能够根据用户查询背后的语义信息，返回更准确、更结构化的信息。Google 知识图谱的宣传语 “things not strings” 道出了知识图谱的精髓：不要无意义的字符串，需要文本背后的对象或事物。

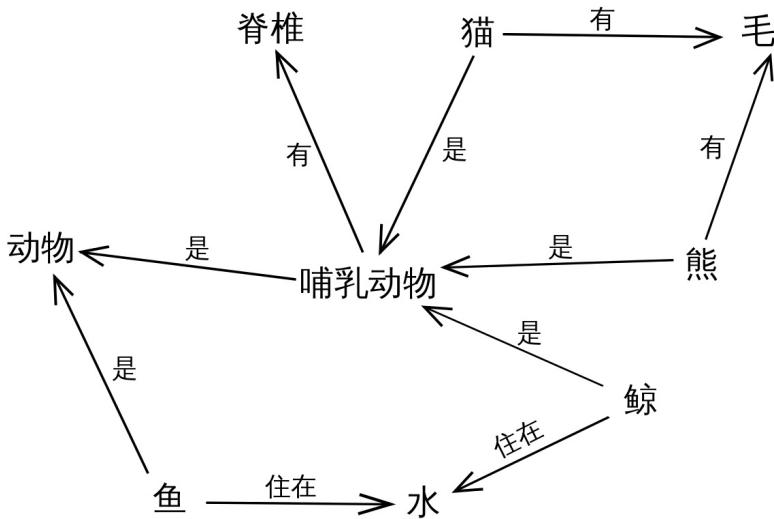
我们可以把知识图谱认为是一个知识库。比如在 Google 搜索引擎里输入 “Who is the wife of Bill Gates?”，我们直接可以得到答案 -“Melinda Gates”。这是因为我们在系统层面上已经创建好了一个包含 “Bill Gates” 和 “Melinda Gates” 的实体以及他们之间关系的知识库。所以，当我们执行搜索的时候，就可以通过关键词提取 (“Bill Gates", "Melinda Gates", "wife") 以及知识库上的匹配可以直接获得最终的答案。这种搜索方式跟传统的搜索引擎是不一样的，一个传统的搜索引擎它返回的是网页、而不是最终的答案。我们只能得到包含这个关键词的网页，然后不得不点击进入相关网页查找需要的信息，所以就多了一层用户自己筛选并过滤信息的过程。



## 知识图谱的前世今生

通过上面这个例子，大家应该对知识图谱有了一个初步的印象，其本质是为了表示知识。

其实知识图谱的概念并不新，它背后的思想可以追溯到上个世纪五六十年代所提出的一种知识表示形式——语义网络 (Semantic Network)。语义网络由相互连接的节点和边组成，节点表示概念或者对象，边表示他们之间的关系 (is-a 关系，比如：猫是一种哺乳动物；part-of 关系，比如：脊椎是哺乳动物的一部分)，如下图。在表现形式上，语义网络和知识图谱相似，但语义网络更侧重于描述概念与概念之间的关系，（有点像生物的层次分类体系——界门纲目科属种），而知识图谱则更偏重于描述实体之间的关联。



## 知识抽取

知识图谱的构建是后续应用的基础，而且构建的前提是需要把数据从不同的数据源中抽取出来。对于垂直领域的知识图谱来说，它们的数据源主要来自两种渠道：一种是业务本身的数据，这部分数据通常包含在公司内的数据库表并以结构化的方式存储；另一种是网络上公开、抓取的数据，这些数据通常是以网页的形式存在所以是非结构化的数据。

前者一般只需要简单预处理即可以作为后续 AI 系统的输入，但后者一般需要借助于自然语言处理等技术来提取出结构化信息。比如在上面的搜索例子里，Bill Gates 和 Malinda Gate 的关系就可以从非结构化数据中提炼出来，比如维基百科等数据源。

## 知识图谱的存储

知识图谱主要有两种存储方式：一种是基于 RDF 的存储；另一种是基于图数据库的存储。它们之间的区别如下图所示。

在知识图谱中我们用 RDF 形式化的表示这种三元关系。RDF，Resource Description Framework，资源描述框架，是 W3C 制定的一种用于描述实体 / 资源的标准数据模型。RDF 中有三种类型：IRI、blank node 和 literals，常用的是 IRI 和 literals。

IRI，International Resource Identifier，可以看做是 URI 或 URL 的泛化，用于在整个知识图谱中唯一的表示一个实体 / 资源。

literals，字面量，可以看做是带有数据类型的纯文本，比如罗纳尔多的全名是 Ronaldo Luís Nazário de Lima 这个字面量可以表示为：

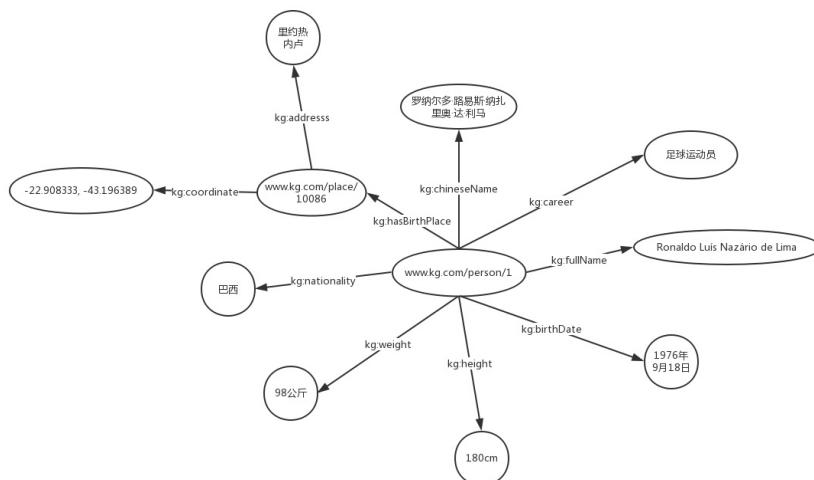
```
Ronaldo Luís Nazário de Lima"^^xsd:string
```

那么，“罗纳尔多的中文名是罗纳尔多 · 路易斯 · 纳扎里奥 · 达 · 利马”这个 SPO 三元组用 RDF 形式表示就是：



```
http://www.kg.com/person/1 kg:fullName "Ronaldo Luís Nazário de Lima"^^xsd:string
```

将上面的知识图用更正式的形式画出来：



© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook 修订时间： 2020-03-13

# 第3章 硬件基础：智能小白

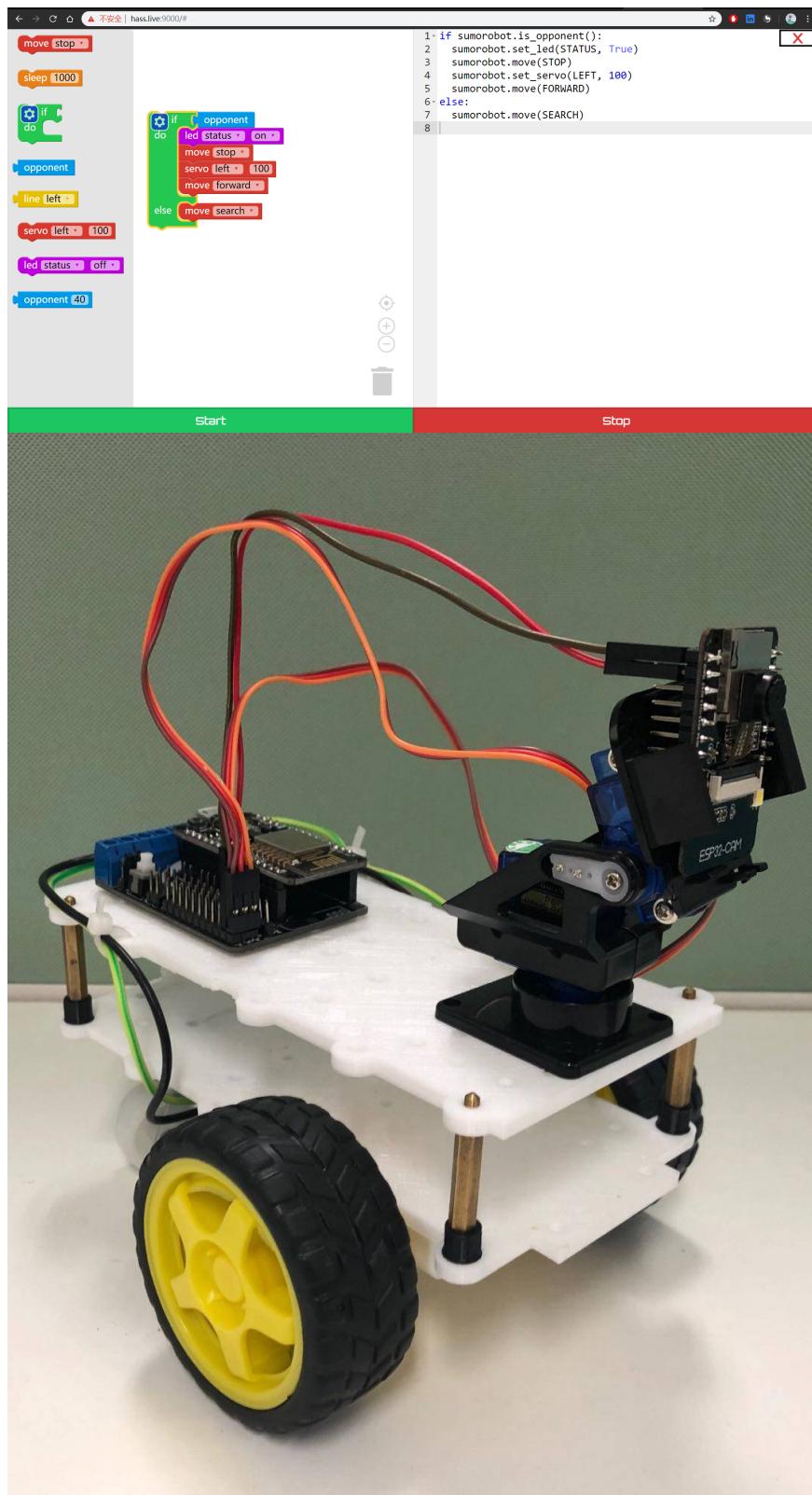
---

## 本章主题：了解硬件操作

主要使用ESP8266提供Web服务，来控制电机、舵机、传感器等，实现远程遥控、控制机械臂抓取、远程视频监控等功能。可以通过积木编程的方式来对小车的功能进行编程。

## 本章重点

- 1 了解开源硬件和电路、网络的基础知识
- 2 掌握一定的通过开源硬件解决实际问题的能力



## 涉及软硬件

- ESP8266, ESP32-CAM
- 小车套件（可3D打印）
- 舵机，灰度传感器、超声波传感器等
- 自行开发的物联网控制平台

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook 修订时间： 2019-12-15

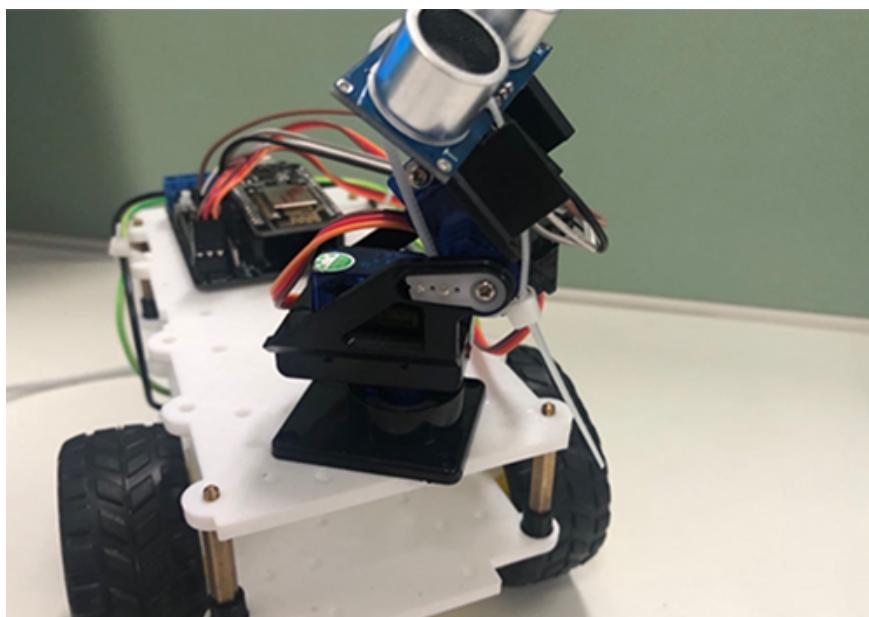
# 第4章 机器视觉：自动追踪小车大白

## 本章主题：机器视觉

从机器视觉出发，让学生理解机器视觉的相关概念和原理，辨别OpenCV和深度学习的异同点。使用OpenCV来处理视觉信号，并通过蓝牙或串口来将处理过的视觉信号发送给小车，从而实现物体追踪，人脸追踪，智能机械臂抓取等功能。学生通过使用Python，完成信息采集：爬虫、多文件处理；信息处理：训练采集的数据，形成分类器，从而让计算机视觉系统能够对特定的物体进行分辨。

## 本章重点

- 了解计算机视觉的相关概念和原理
- 了解OpenCV和深度学习的关系和区别
- 了解Python在图像处理方面的一些基本操作
- 了解模式识别，会训练分类器，并使用开源硬件来对图像处理结果做简单的反馈



## 涉及软硬件

- 树莓派、Arduino
- 舵机、CSI摄像头、电磁传感器
- 小车套件
- 3D打印机

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间：2019-12-15

## 第2节 大白智能分拣

使用OpenCV来识别圆形物体，若识别到则发送串口指令给Arduino。

Arduino控制机械臂和电磁铁进行抓取

### 训练大白认识圆形

利用霍夫变换来进行圆环检测。一个圆环需要3个参数来确定，所以进行圆环检测的累加器必须是三维的，这样效率就会很低，因此OpenCV使用了霍夫梯度法这个巧妙的方法，来使用边界的梯度信息，从而提升计算的效率。

OpenCV中进行霍夫圆环检测的函数：

```
cv2.HoughCircles(image, method, dp, minDist, circles=None, param1=None, param2=None, n
```

本例中使用的具体参数：

```
cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT , 1, 100, param1=100, param2=100, minRadius=
```

参数解释：**image**: 8位，单通道图像。如果使用彩色图像，需要先转换为灰度图像。

**method**: 定义检测图像中圆的方法。目前唯一实现的方法是  
cv2.HOUGH\_GRADIENT。

**dp**: 累加器分辨率与图像分辨率的反比。**dp**获取越大，累加器数组越小。例如，如果**dp=1**时，累加器和输入图像具有相同的分辨率。如果**dp=2**，累加器便有输入图像一半那么大的宽度和高度。

**minDist**: 为霍夫变换检测到的圆的圆心之间的最小距离，即让我们的算法能明显区分的两个不同圆之间的最小距离。这个参数如果太小的话，多个相邻的圆可能被错误地检测成了一个重合的圆。反之，这个参数设置太大的话，某些圆就不能被检测出来了。

**param1**: 有默认值100。它是**method**设置的检测方法的对应的参数。对当前唯一的方法霍夫梯度法，它表示传递给canny边缘检测算子的高阈值，而低阈值为高阈值的一半。

**param2**: 也有默认值100。它是**method**设置的检测方法的对应的参数。对当前唯一的方法霍夫梯度法，它表示在检测阶段圆心的累加器阈值。它越小的话，就可以检测到更多根本不存在的圆，而它越大的话，能通过检测的圆就更加接近完美的圆形了。

**minRadius**: 默认值0，表示圆半径的最小值。单位是像素。

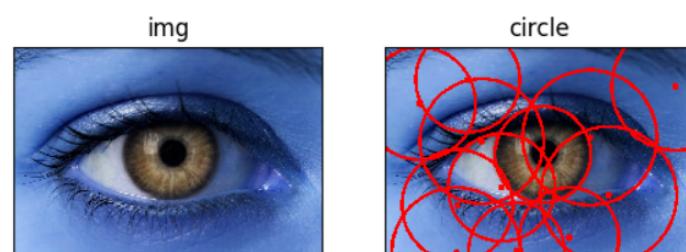
**maxRadius**: 也有默认值0，表示圆半径的最大值。单位是像素。

示例：

原始图像:



处理后图像:



## 会自动抓取硬币的机械臂

视觉图像经由OpenCV处理后，如果识别到圆形，就通过串口将信息发送给Arduino，Arduino接收到信号后，控制云台舵机转动，并通过电磁铁将圆形物品分拣处理

## 原理图

```
摄像头\n机械臂\n电磁传感器->大白\n树莓派: CSI/USB/GPIO
大白\n树莓派-->传送带: 摄像头实时监测
Note right of 大白\n树莓派: OpenCV处理摄像头数据\n发送指令给机械臂和电磁传感器
```

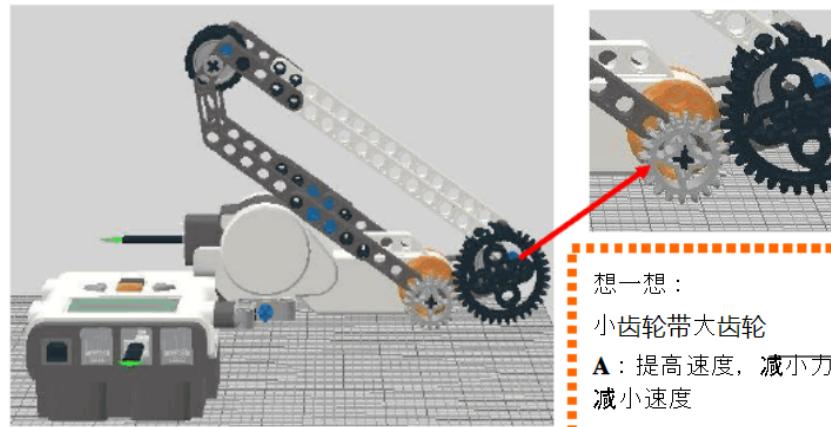
## 硬件准备

- 大白
- Arduino
- CSI摄像头
- 摄像头桌面支架
- 待检测的圆形金属物体（任选）

## 硬件连接

- 将烧录好程序的Arduino通过USB连接到树莓派
- 电磁传感器连接到pin5，两个舵机连接到3和4
- 用乐高制作传送带和摄像头支架[参考活动](#)，将待检测的物体放置在传送带上，使传送带缓慢转动
- USB摄像头垂直固定在传送带顶部，使之能够看到传送带上的物体

### 2. 设计与搭建



想一想：  
小齿轮带大齿轮  
A: 提高速度, 减小力量  
减小速度

## 启动智能分拣系统

1. 使用远程桌面或HDMI视频输出连接到树莓派
2. 打开终端，输入 `cd ~/Desktop/learn-ai/codes/chapter4/part2_AutoSort/AutoSort`
3. 输入 `python classifier.py`

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook修订时间：2019-12-15

## 第3节 大白自动追踪

通过拍照、爬虫等方式获取待训练图片，使用python进行简单的文件处理，用OpenCV训练后，可实现对特定物体的识别和追踪

### 级联分类器简介

目前图像检测方法主要分为两大类，基于知识和基于统计。

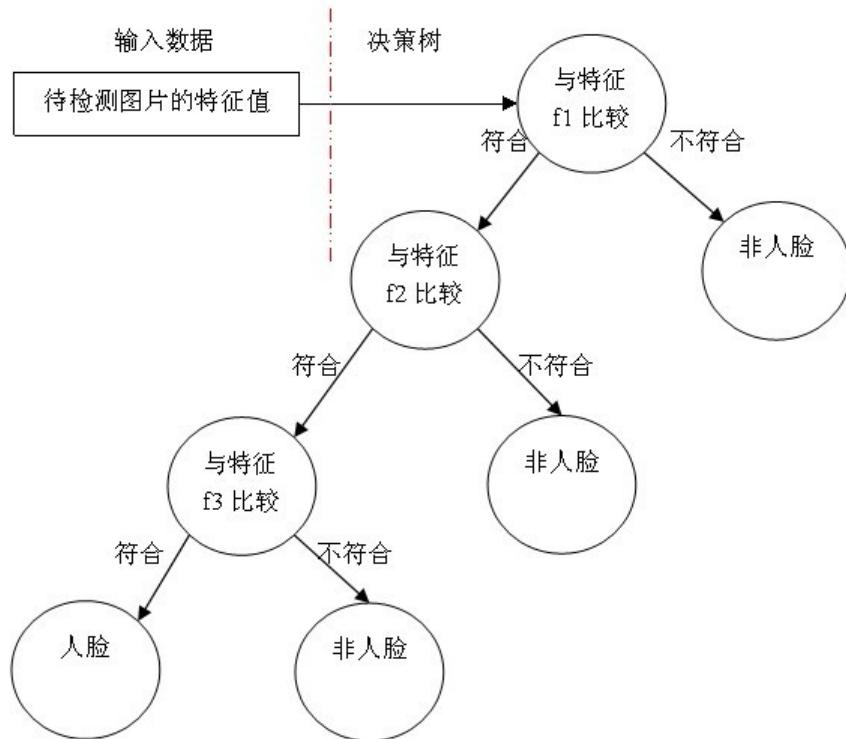
以人脸检测来说，基于知识的人脸检测方法主要包括：模板匹配，人脸特征，形状与边缘，纹理特征，颜色特征。

基于统计的人脸检测方法主要包括：主成分分析与特征脸法，神经网络模型，隐马尔可夫模型，支持向量机，**Adaboost**算法。

基于知识的方法将人脸看成不同特征的特定组合，即通过人脸的眼睛、嘴巴、鼻子、耳朵等特征及其组合关系来检测人脸。

基于统计的方法将人脸看成统一的二维像素矩阵，通过大量的样本构建人脸子空间，通过相似度的大小来判断人脸是否存在。

OpenCV采用的级联分类器，可以理解为将采用**Adaboost**算法构建的若干分类器串联起来，即级联。只有通过所有分类器后才识别为检测正确。这样可以提高检测的准确度。



OpenCV提供了若干已经提前训练好的级联分类器供使用，包括人体、人脸、猫咪等。

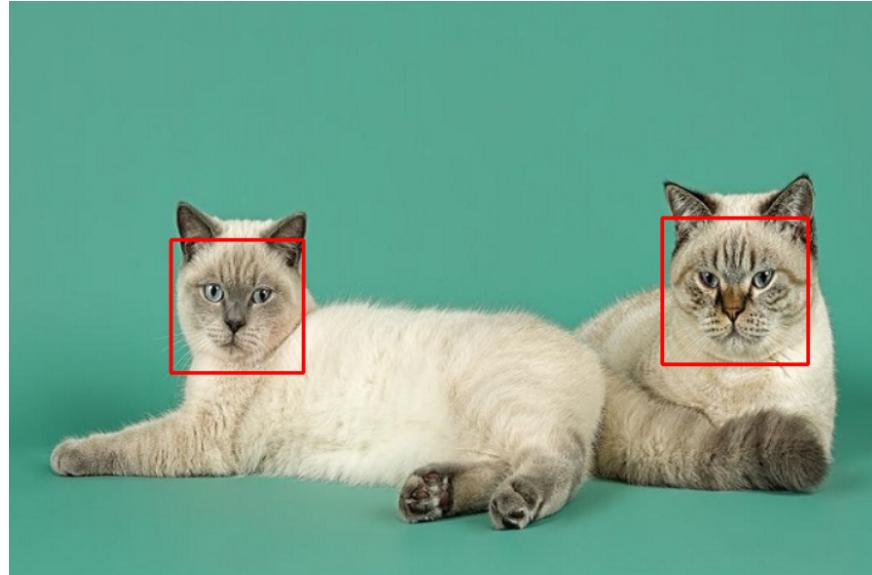


### 示例：识别猫脸

核心代码：

```
String catFileName = "~opencv/build/etc/haarcascades/haarcascade_frontalcatface.xml"
CascadeClassifier catclassifier
```

结果：



### 活动1：人脸分类器：追着人跑的大白

#### 原理图

```
摄像头->大白\n树莓派：CSI/USB
大白\n树莓派-->Arduino：串口指令
Note right of 大白\n树莓派：OpenCV处理摄像头数据（级联分类器）\n发送指令给Arduino
```

#### 硬件准备

- 大白车
- Arduino
- CSI摄像头

- 待识别的人或猫的照片（电子或纸质打印任选）

## 硬件连接

- 将烧录好程序的Arduino通过USB连接到树莓派

## 启动追踪小车

- 使用远程桌面或HDMI视频输出连接到树莓派
- 打开终端，输入 `cd ~/Desktop/learn-ai/codes/chapter4/part3_AutoTrack/AutoTrack`
- 输入 `python tracker.py`
- 将待识别的物体（人脸或猫）在车前移动，观察车的追踪行为

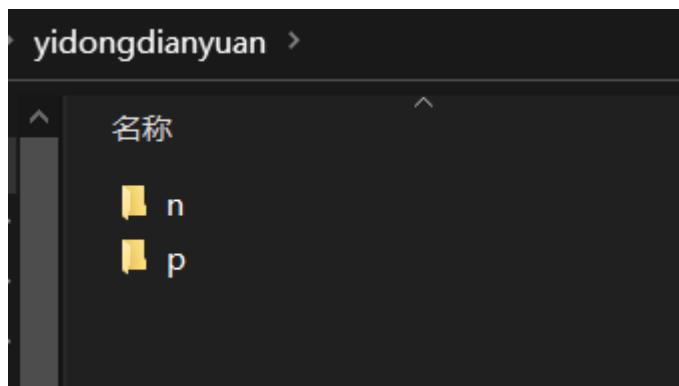
## 活动2：训练新的分类器

通过收集大量样本图片，可以训练自定义的分类器，可以识别任意的物体。

### 1.环境准备

在Windows桌面上执行：

- 新建一个文件夹，重命名为待检测的物体名称（英文），比如检测移动电源，就命名为yidongdianyuan
- 进入文件夹，再新建两个文件夹，分别命名为 p 和 n，p 代表 positive（正样本），n 代表 negative（负样本）

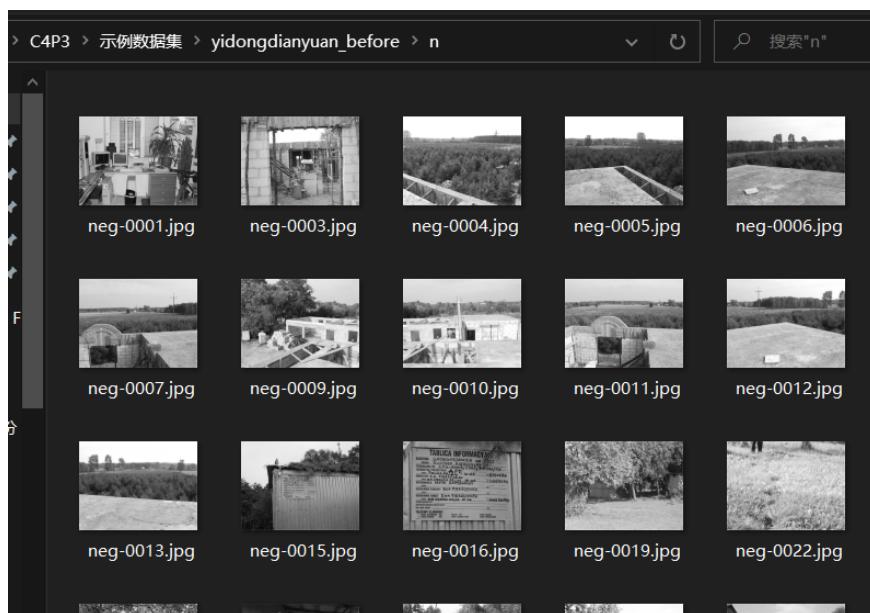


- 获取分类器图形化训练软件，下载安装。
- 获取图片批量处理软件，解压即可，主程序为 `MtPcl.exe`
- 获取示例数据集

### 2.收集负样本

训练样本包括正样本和负样本。正样本，通俗点说，就是图片中只有你需要的目标。而负样本的图片只要其中不含有目标就可以了。但需要说明的是，负样本也并非随便选取的。例如，需要检测的目标是汽车，那么正样本就应该是仅仅含有汽车的图片，而负样本显然不能是一些包含天空的，海洋的，风景的图片。因为最终训练分类器的目的是检测汽车，而汽车应该出现在马路上。也就是说，分类器最终检测的图片应该是那些包含马路，交通标志，建筑物，广告牌，汽车，摩托车，三轮车，行人，自行车等在内的图片。很明显，这里的负样本应该是包含摩托车、三轮车、自行车、行人、路面、灌木丛、花草、交通标志、广告牌等。**Adaboost**方法是机器学习中的一个经典算法，而机器学习算法的前提条件是，测试样本和训练样本独立同分布。所谓的独立同分布，可以简单理解为：训练样本要和最终的应用场合非常接近或者一致。否则，基于机器学习的算法并不能保证算法的有效性。此外，足够的训练样本（至少得几千张正样本、几千张负样本）也是保证训练算法有效性的一个前提条件。

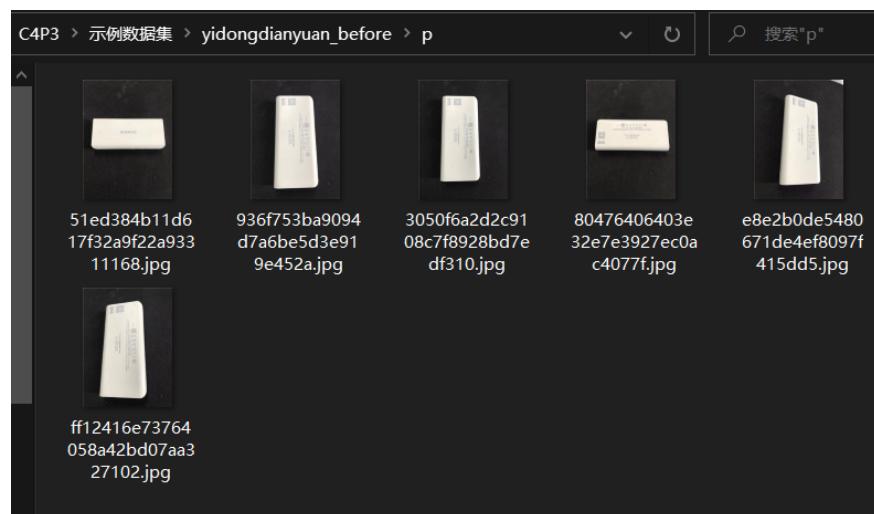
使用网络浏览器或各种方法，收集各种图片，但是不能包含待检测的物体（移动电源）。保存在 **n** 文件夹中。数量为数十个为宜。



### 3. 收集正样本

正样本就是想要识别出来的物体。尽可能排除无关物体的干扰（图片中无其他物体）

使用手机，拍摄待识别的物体，并将其存储在 **p** 文件夹。数量在10个以上为宜。



#### 4. 图片预处理

这一步骤调整正负样本的图片分辨率。

- 打开 MtPcl.exe , 选择 添加文件夹 , 选择 p 文件夹



- 选择右侧 修改尺寸 , 高度设置为480, 勾选 保持原图比例 。然后选择下面的 覆盖原图 。点击保存。



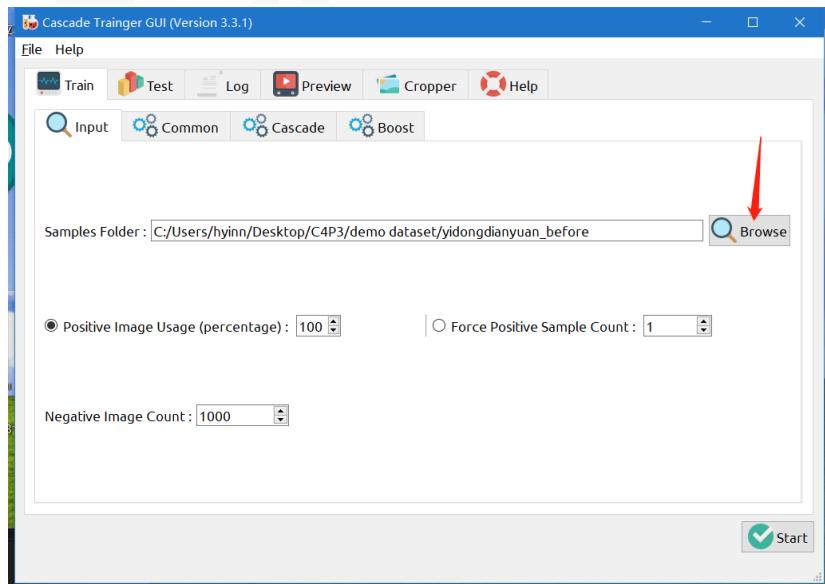
- 点击 清空，并对 n 文件夹执行相同的操作



## 5.训练分类器

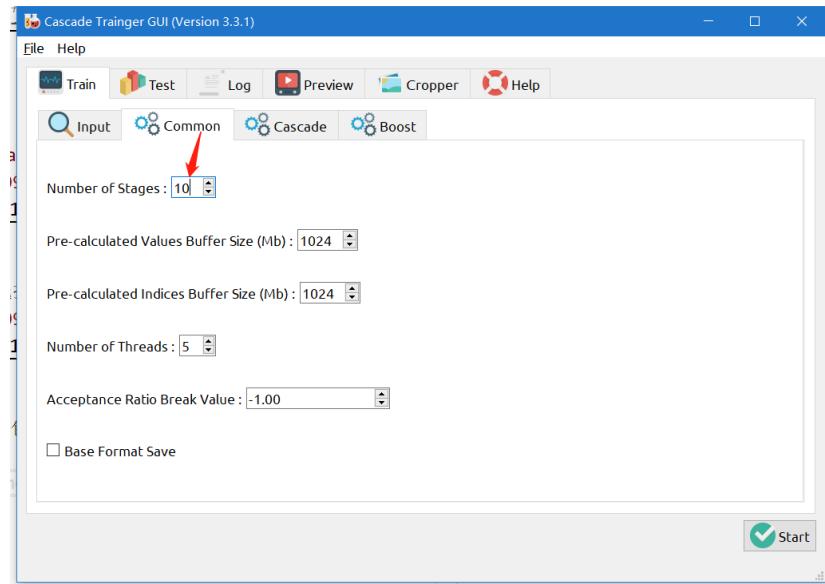


- 打开 Cascade-Trainer-GUI
- 点击 Browse，选择包含 n 和 p 的文件夹

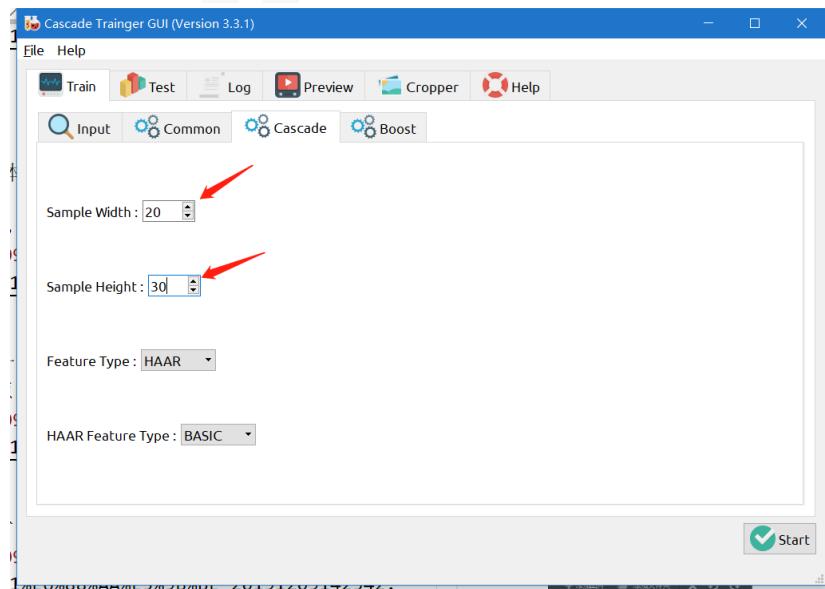


路径中不能包含中文

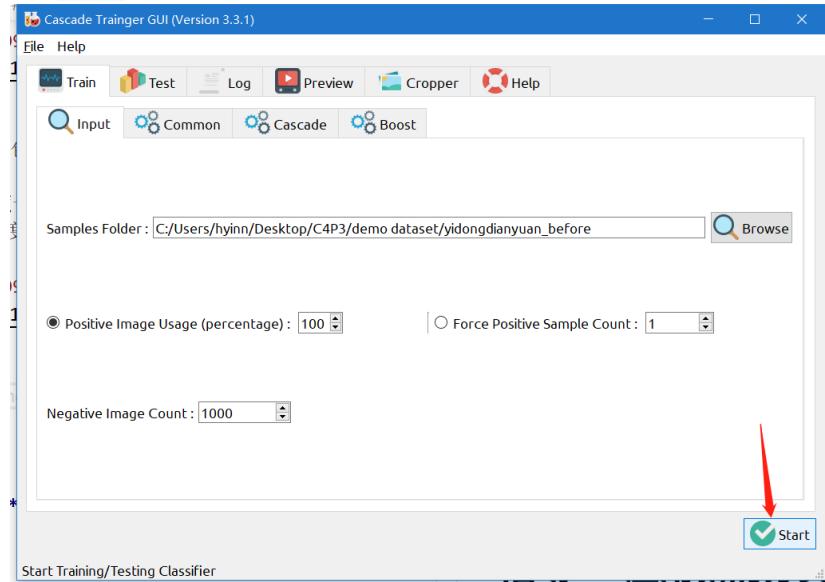
- 单击 Common 选项卡，调整第一项 Number of Stages，选择10获得更快的训练速度，如果效果不明显可以逐渐增大，但是训练速度会显著增加。



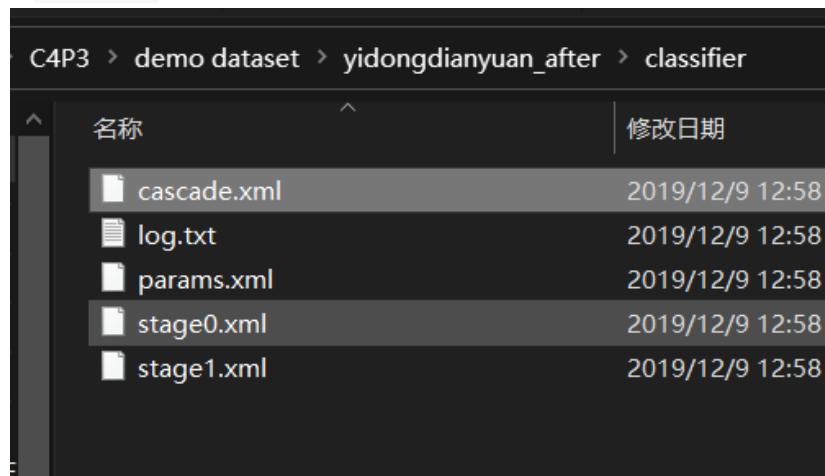
- 修改宽度和高度为 20， 30。使宽高比和待处理的文件相同。



- 点击 Start 开始训练



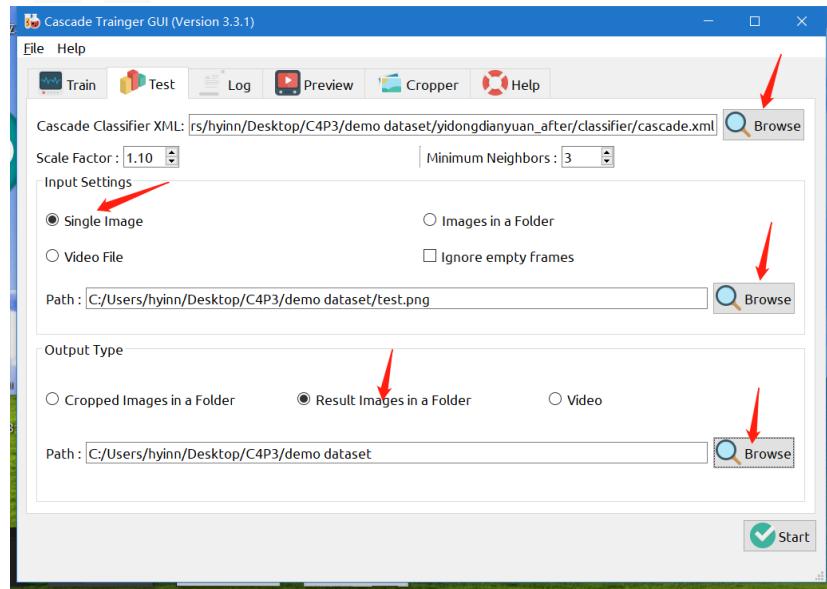
- 训练成功后，在原文件夹中会增加一个 `classifier` 文件夹，里面的 `cascade.xml` 就是训练成功的级联分类器。



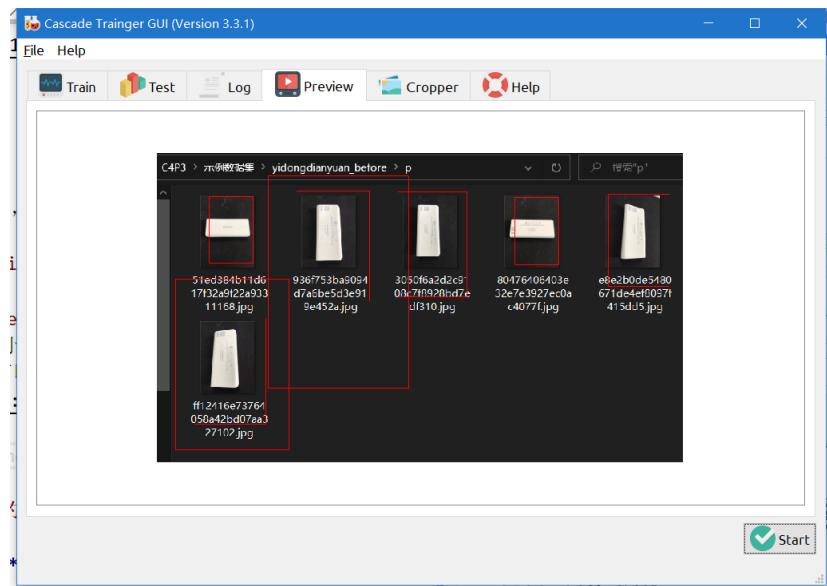
## 6. 测试

- 选择 `Test` 选项卡，点击右上角 `Browse`，选择上一步的 `cascade.xml`
- 输入设置，选择 `Single Image`，测试图片可以是包含多个目标物体的图片或截图
- 选择输出方式为 `Result Image in a Folder`，并指定路径

- 点击 Start , 最小检测阈值输入 10 , 10 ; 最大检测阈值输入 800 , 800 (可以不断调整以取得最好的效果)



- 检测效果



### 活动3：应用新的分类器

将 `cascade.xml` 通过U盘拷贝到树莓派的路径下

名称	修改日期
tracker_arduino	2019/8/30 11:08
cascade.xml	2019/12/9 12:58
haarcascade_frontalface_default.xml	2019/8/6 15:09
tracker.py	2019/9/27 16:48
tracker_my_object.py	2019/12/9 14:55

打开终端，执行：

```
cd ~/Desktop/learn-ai/codes/chapter4/part3_AutoTrack/AutoTrack  
python tracker_my_object.py
```

大白将会跟随训练的物体进行移动。

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2019-12-15

# 第5章 深度学习：无人驾驶小车小白

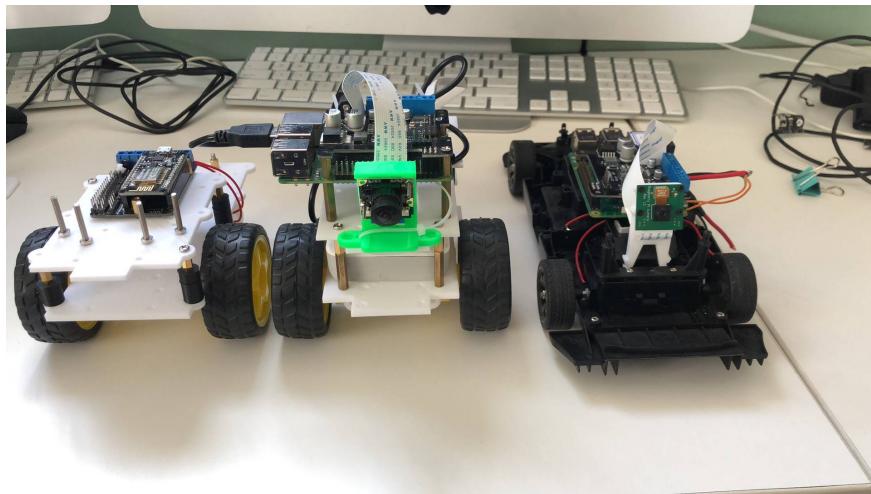
## 本章主题：深度学习、无人驾驶

这部分基于树莓派以及一些开源软件构建。树莓派从摄像头模块获取输入，然后通过无线方式发送获得的图像数据到电脑，电脑通过之前训练好的神经网络对输入的图像数据预测小车接下来的动作，然后发送这些预测动作的控制指令到树莓派控制小车的程序中。小车根据这些获得的指令实现自动驾驶。

现有的Caffe、TensorFlow等工具箱已经很好地实现CNN模型，但这些工具箱需要的硬件资源比较多，不利于初学者实践和理解。本章使用NumPy来构建卷积神经网络（Convolutional Neural Network,CNN）模型，通过对驾驶数据的采集和训练，实现无人驾驶。

## 本章重点

- 1 掌握无人驾驶数据采集及训练的基本方法
- 2 会灵活地在无人驾驶系统中训练和应用分类器



## 涉及软硬件

- 树莓派、摄像头
- 小车套件
- OpenCV

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间：2019-12-15

## 第2节 无人驾驶数据采集、训练与测试

搭建起车道，然后运行相应的收集数据的程序，按下键盘方向键控制小车行驶，每按一次方向键，程序就会记录下一帧相应的图像。让小车平均遍历自动驾驶中可能出现的各种情况，按‘q’退出数据采集，然后再运行相应的模型训练程序训练自动驾驶神经网络。最后使用训练好的神经网络模型在跑道上进行测试。

### 原理图

摄像头->大白\n树莓派：CSI/USB  
大白\n树莓派-->客户端：WiFi  
Note right of 大白\n树莓派：HTTP协议传送摄像头数据\n神经网络收集和训练数据

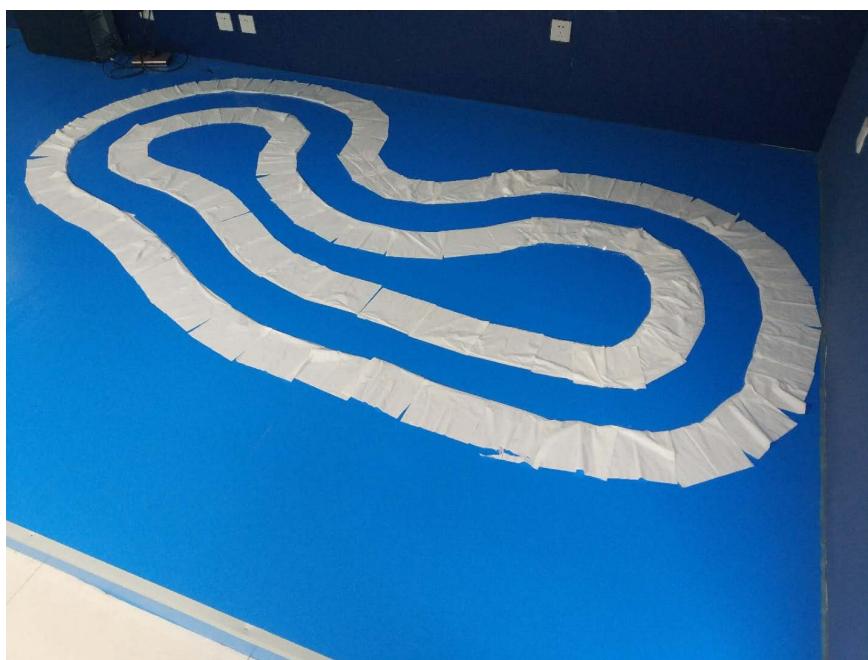
### 硬件准备

#### 硬件清单

- 纸
- 胶带

#### 硬件搭建-跑道

- 地面颜色为纯色，与所用纸张的颜色对比度应较大
- 跑道的宽度稍大于车的宽度
- 可以把拐弯处的弯度设计得稍大一些



## 采集驾驶数据

### 1. 打开终端，执行以下命令

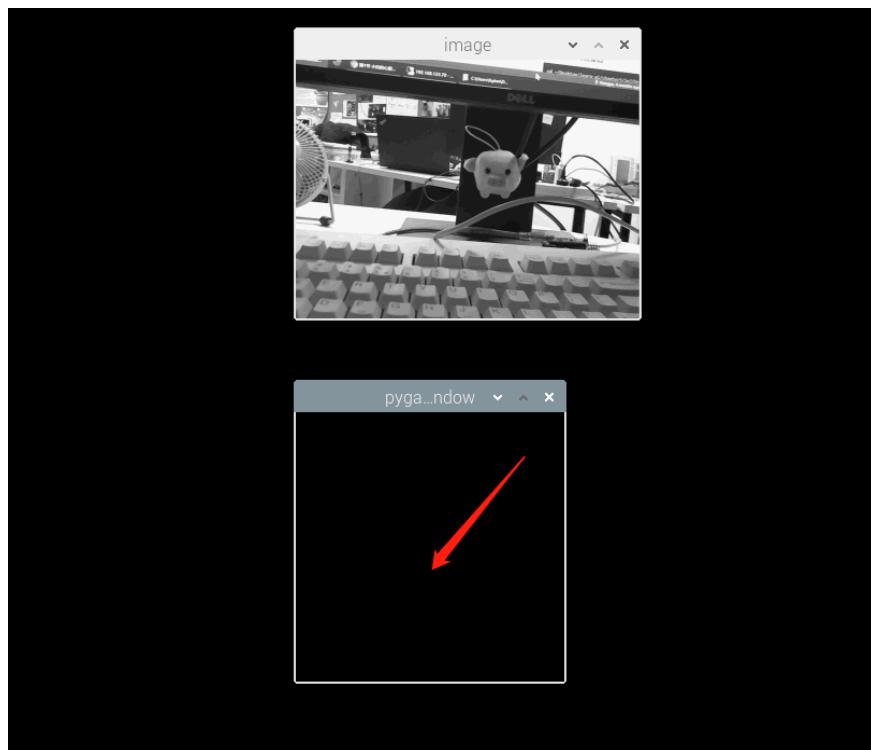
```
cd ~/Desktop/learn-ai/codes/chapter5/SelfDrivingCar  
cd computer  
python3 collect_training_data.py
```

### 2. 新建一个终端窗口

```
cd ~/Desktop/learn-ai/codes/chapter5/SelfDrivingCar  
cd raspberryPi  
python3 stream_client.py
```

### 3. 开始采集

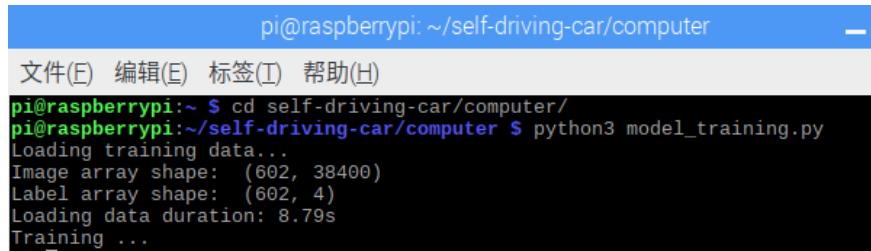
- 顺利执行后会出现两个窗口，上面的是摄像头的画面，下面的是操作区。将鼠标焦点移到箭头所指的工作区上。
- 把小车放置在跑道上，点击键盘上下左右光标控制小车。
- 通过键盘控制，让小车在跑道上正确的绕行数圈（3圈左右即可）
- 训练结束后，确定焦点仍在工作区上，点击键盘 **q** 退出训练，程序会自动保存驾驶数据



## 训练驾驶数据

### 1. 新建一个终端窗口

```
cd ~/Desktop/learn-ai/codes/chapter5/SelfDrivingCar  
cd computer  
python3 model_training.py
```



```
pi@raspberrypi:~$ cd self-driving-car/computer/  
pi@raspberrypi:~/self-driving-car/computer $ python3 model_training.py  
Loading training data...  
Image array shape: (602, 38400)  
Label array shape: (602, 4)  
Loading data duration: 8.79s  
Training ...
```

## 2.得到模型

模型文件在 `~/Desktop/learn-ai/chapter5/SelfDrivingCar/computer/saved_model/nn_model.xml`

## 开始无人驾驶

根据训练好的神经网络模型，现在我们可以实现自动驾驶

### 1.打开终端

```
cd ~/Desktop/learn-ai/codes/chapter5/SelfDrivingCar  
cd computer  
python3 rc_drive_nn_only.py
```

### 2.新建一个终端窗口

```
cd ~/Desktop/learn-ai/codes/chapter5/SelfDrivingCar  
cd raspberryPi  
python3 stream_client.py
```

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2019-12-15

# 第6章 综合进阶：机器人小绿

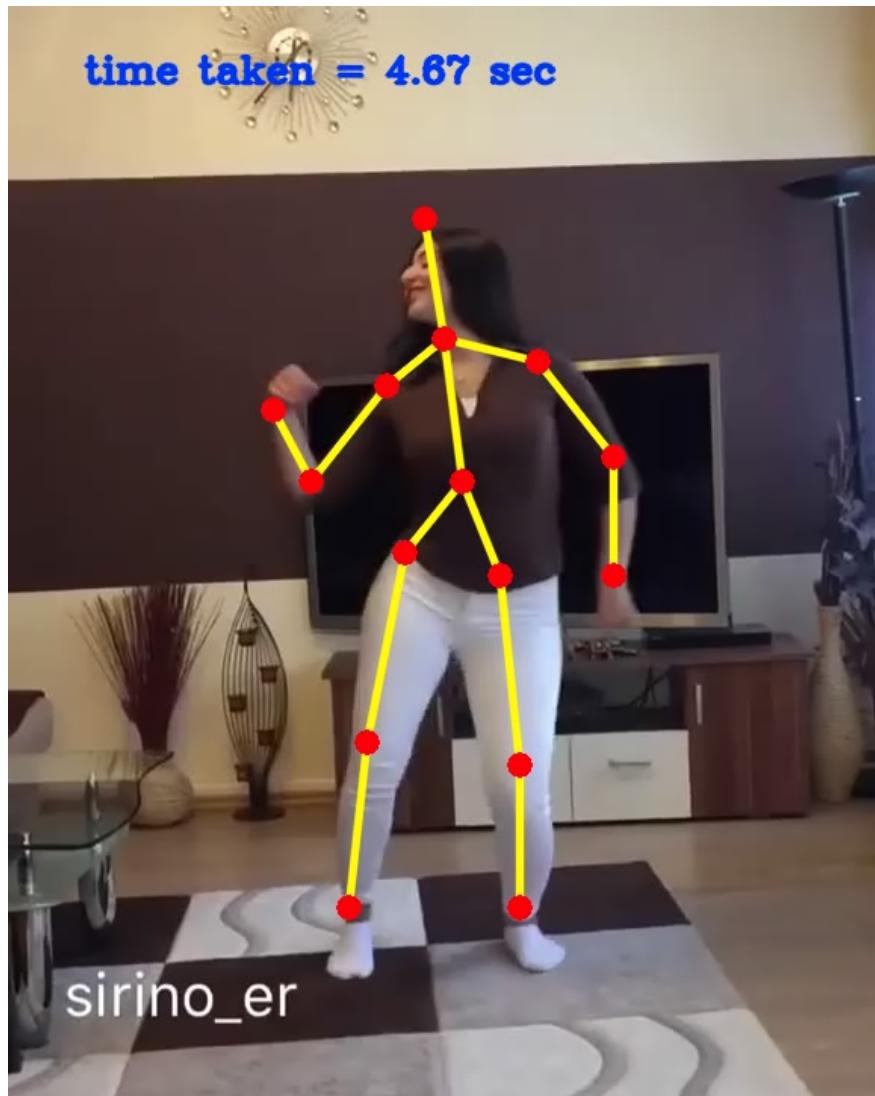
## 本章主题：机器人

小绿是一个使用3D打印制作外壳，使用舵机作为动力部分，使用树莓派作为控制中枢的智能机器人。作为物联网的一个节点，实现多种物联网功能，包括网页遥控：通过自行开发的物联网平台来对它进行遥控；语音助手：可以通过自己训练的热词来进行唤醒、通过语音来控制机器人执行各种动作；控制其他设备：比如控制前几个章节的小车，读取各种传感器的数据等；人脸解锁：通过实时的人脸识别和红外线发射装置，实现人脸解锁，也可以通过Google Assistant、Siri、Alexa等远程控制；实时姿态模仿：通过单目摄像头拍摄实时画面，采用OpenPose姿态识别软件进行处理，将关节姿态数据通过蓝牙或串口传递给机器人，机器人进行实时的姿态模仿。

## 本章重点

- 1 了解物联网的基本概念和原理
- 2 会通过物联网和开源硬件制作较为复杂的综合机器人系统





## 涉及软硬件

- 树莓派、ESP8266、安卓手机
- 麦克风阵列、舵机、摄像头
- 3D打印机
- OpenPose、OpenCV

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2019-12-15

## 第2节 小绿的一小步，我们的一大步

小绿通过移动电源即可供电。接通电源后，访问小绿的ip地址，试着让小绿迈出第一步吧

### 原理图

小绿->手机浏览器：1.建立局域网服务器  
手机浏览器-->小绿：2.通过WiFi访问控制界面  
手机浏览器-->小绿：3.发送请求给服务器，如“前进”  
小绿->手机浏览器：4.服务器响应请求，让小绿的动力系统执行

### 组装小绿

小绿的外壳是3D打印而成。将舵机固定在关节处。然后将舵机都接在主控板上就可以了。是不是很简单呢？小绿的3D打印源文件在 `learn-ai/assets/3D Models/green`

- 小绿共需要9个舵机，每个胳膊2个共4个，每条腿2个共4个，还有1个在颈部。
- 按顺序将舵机用螺丝刀固定在3D打印件上，完成组装。注意将舵机的线都引向中间。

### 烧录程序到开发板（选做）

程序烧录过程略，程序源文件见 `learn-ai/codes/chapter6/part2_FirstStep/greenrobot`

### 小绿迈出第一步

#### 1.将小绿连接到移动电源

#### 2.查看ip地址

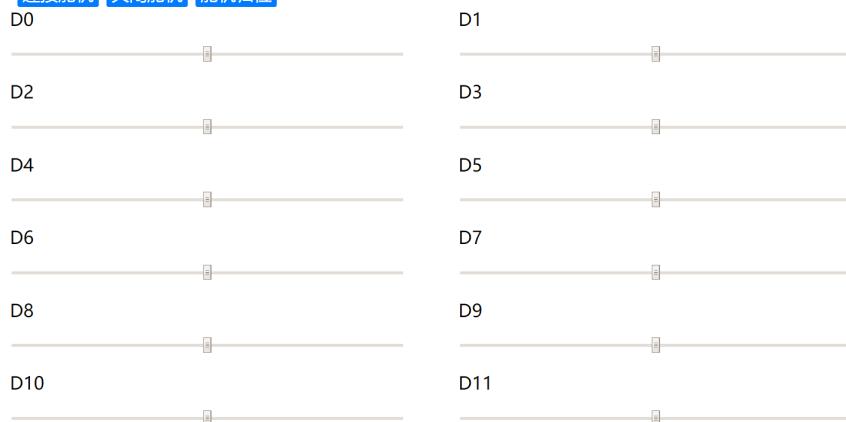
使用浏览器打开[路由器管理地址](#)，查找小绿的ip地址

#### 3.访问测试地址

在浏览器中打开 小绿ip地址/test

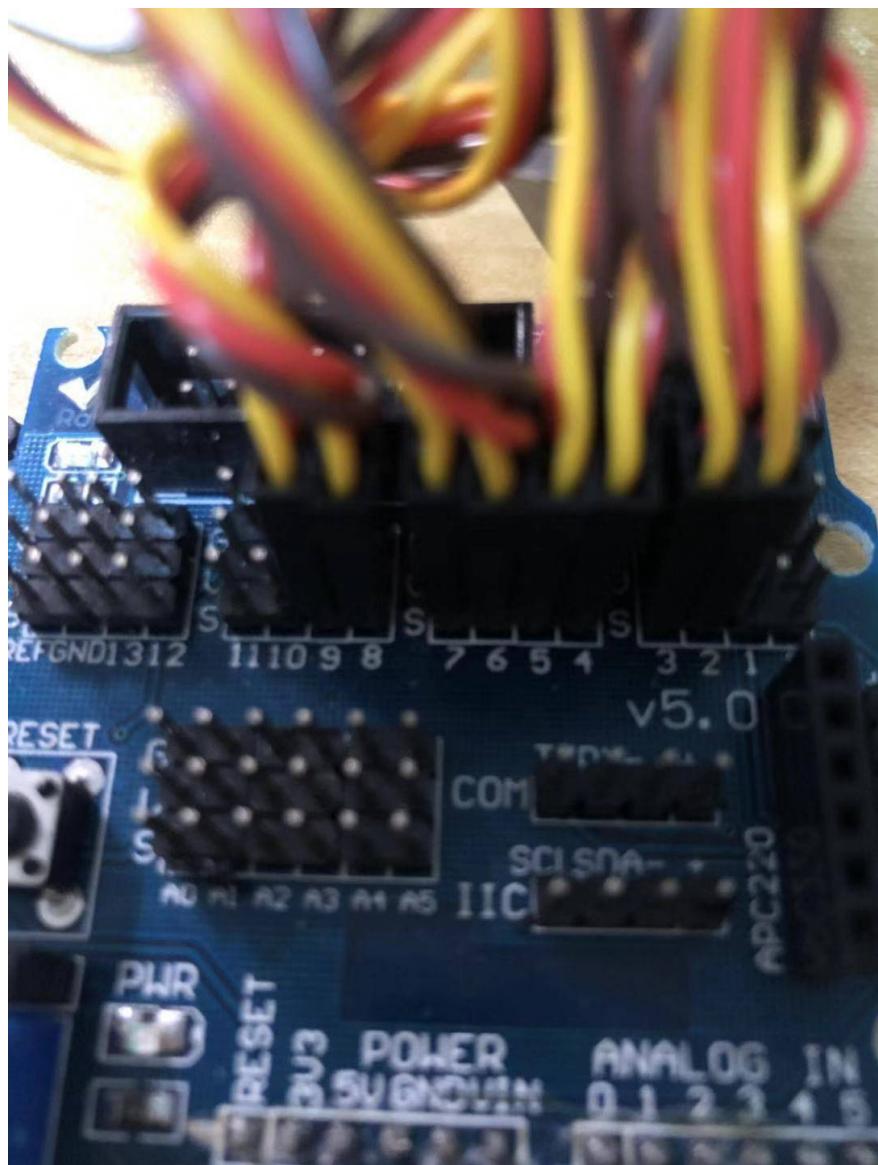
## 小绿机器人测试面板

[连接舵机](#) [关闭舵机](#) [舵机归位](#)

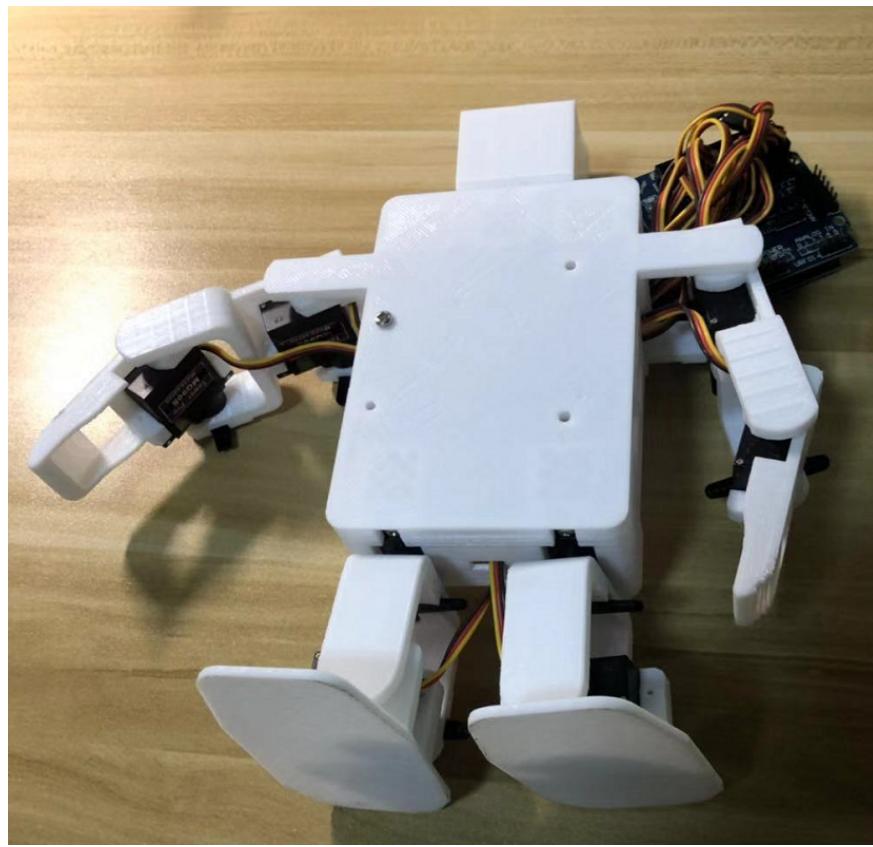


## 4. 正确连接舵机

逐个将舵机的线连接到基于ESP8266芯片的Wemos D1开发板扩展板上。从0到11共12个槽位都可以。按照下面的对应关系将小绿的舵机连接到正确的槽位上，并进行测试。



小绿的不同部位与槽位的对应关系如下：此面为正面



位置	槽位	位置	槽位
左手	D5	右手	D4
左脚	D9	右肩	D2
左腿	D7	右腿	D6
左肩	D3	右脚	D8

## 5. 访问 小绿ip地址

最后，终于可以让小绿迈出第一步了！

### 小绿机器人控制面板



使用**blockly**积木控制小绿

打开 `learn-ai/codes/chapter6/part2_FirstStep/robot_diy_blockly/index.html`，通过积木拖拽控制小绿



© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook  
修订时间： 2019-12-15

## 第4节 使用物联网制作人脸解锁

### 原理图

```
摄像头-->树莓派: WiFi  
Note right of 树莓派: HomeAssistant处理人脸数据匹配  
树莓派-->语音合成: 播放识别结果  
树莓派-->红外发射器: 发送处理结果  
红外发射器-->大门\n电灯\n空调\n电视\n.....: 红外信号
```

### 硬件清单

- 树莓派
- 安卓手机
- 红外发射器
- 具有红外遥控器的大门/灯/空调/电视/.....

### 硬件准备

#### 摄像头和扬声器

- 在安卓手机上下载安装 IP摄像头 和 Kodi 两个app
- 将安卓手机连接到教学WiFi，通过路由器查询记录手机的ip地址
- 打开 IP摄像头 app，进行设置后点击最下面的 开启服务器，记录下视频服务的地址和端口
- Kodi是为了通过安卓手机的扬声器远程播放声音

#### 红外发射器

- 安卓手机安装 智慧家 app，对博联RM Pro红外发射器进行初始化设置，连接到教学WiFi，然后通过路由器查询记录ip地址和mac地址

#### HomeAssistant的启动

##### 1.启动HomeAssistant

```
sudo docker start home-assistant 启动大概需要1分钟
```

##### 2.访问HomeAssistant

```
树莓派ip:8123，选择API密码登陆，密码 welcome
```

##### 配置HomeAssistant（configuration.yaml）

```
sudo docker exec -it home-assistant env LANG=C.UTF-8 /bin/bash  
vi configuration.yaml  
# 按insert键，然后进行输入  
# 退出vi编辑器先按esc键，然后输入:wq，回车  
# 退出bash环境输入exit，回车
```

## 1.基础配置

```
group: !include groups.yaml  
automation: !include automations.yaml  
script: !include scripts.yaml  
scene: !include scenes.yaml  
  
# 如果希望能够从iOS或macOS中的家庭应用来管理，增加下面这一行。  
homekit:  
  
.....  
.....  
.....
```

## 2.Kodi和摄像头配置

```
.....  
.....  
.....  
  
# 在host处填入安卓平板的ip地址  
media_player:  
- platform: kodi  
  host: 192.168.123.194  
  
# android_ip_webcam:  
#   - host: 192.168.123.194  
#     port: 8090  
  
# ffmpeg:  
# camera:  
# - platform: ffmpeg  
#   name: Camera  
#   input: -rtsp_transport tcp -i rtsp://192.168.123.29:8554/live  
# camera:  
#   - platform: rpi_camera  
  
# camera:  
#   - platform: local_file  
#     name: camera01  
#     file_path: /share/motion/lastsnap.jpg  
  
camera:  
- platform: mjpeg  
  mjpeg_url: http://192.168.123.218:8080/?action=stream  
  # mjpeg_url: http://192.168.123.59:8088/?action=snapshot
```

## 3.红外发射配置

```

.....
.....
.....

# 在host和mac处填入红外发射器的ip和mac地址
switch:
  - platform: broadlink
    host: 192.168.123.107
    mac: '78:0F:77:5A:26:85'
    timeout: 15
  switches:
    door:
      friendly_name: "大门"
      command_on: 'eAY0AC8PEAAC2xAuLw8QLi8PEC4vDxEuLw8QLi8QEC4vEBEuLw8vDy8QEC4vDxEuL
      command_off: 'eAY0AC8PEAAC2xAuLw8QLi8PEC4vDxEuLw8QLi8QEC4vEBEuLw8vDy8QEC4vDxEuL

```

- 访问<http://HomeAssistant的ip地址:8123>
- 点击左下角的第一个 服务 按钮，然后选择 `switch.broadlink_learn_command`
- 点击 `Call Service`，这时博联红外发射器的前端会出现小红点。用遥控器要学习的按键对着小红点按下去，小红点消失

## 开发者工具

The service dev tool allows you to call any available service in Home Assistant.

**Services**

- recorder.purge
- script.reload
- script.toggle
- script.turn\_off
- script.turn\_on
- switch.broadlink\_learn\_command\_192\_168\_123\_107**
- switch.broadlink\_send\_packet\_192\_168\_123\_107
- switch.toggle

- 点击左下角的第二个 状态 按钮，找到类似下图的内容，将最后的一长串代码复制到配置文件的对应位置即可

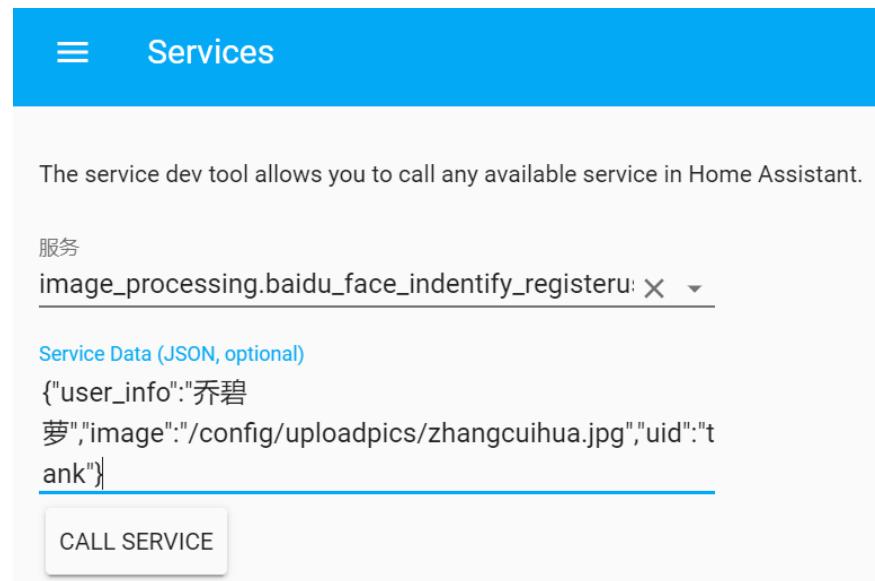
<code>persistent_notification.notification_2</code>	notifying	title: Broadlink switch message: Received packet is: JgBoAMb2EXIRcxFzETERcxExETERMRFzETERMBExETERcxEwEXMRcxFyEXMRchFzE RMBExEXMRMFyETERcxEwETERcxExExIRMRFzETARcxFzEfYRAAOF
---	-----------	--

## 4.人脸识别和tts配置

```
....  
....  
....  
  
# 正确填写下面的参数  
# 百度人脸识别注册: https://cloud.baidu.com/product/face  
sensor:  
  - platform: baidu_face  
    api_key: "tHjWWiNXlQLFNT2SdrNPWwH3"  
    secret_key: "LXHQ5kP6GYewzOqFL1umrK4mf1jx3W4r"  
    group_list: "[normal_group]"  
    camera_entity_id: "camera.mjjpeg_camera"  
    token: "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiI0NjBjMjF1M2NiZjY0YTliYTdjZ  
    # liveliness: "NORMAL"  
    # name: "ren lian shi bie"  
    # port: 8123  
    # pic_url: "网络、本地图片地址"  
    scan_interval: 1  
  
    # image_processing:  
      - platform: baidu_face_indentify  
        app_id: '11478116'  
        api_key: 'tHjWWiNXlQLFNT2SdrNPWwH3'  
        secret_key: 'LXHQ5kP6GYewzOqFL1umrK4mf1jx3W4r'  
        snapshot_filepath: '/home/pi/images/'  
        resize: 0  
        detect_top_num: 3  
        ha_url: 'http://192.168.123.201:8123'  
        # ha_password: 'welcome'  
        scan_interval: 1  
        source:  
          - entity_id: camera.mjjpeg_camera  
            name: faceRec  
  
  
# 百度TTS注册: https://cloud.baidu.com/product/speech/tts  
tts:  
  - platform: baidu  
    app_id: 9217941  
    api_key: qW5HLj4Ks6DfsCV2K9If5080  
    secret_key: 37ricUCmGj1lfrhaGcyu11wWqCjvZbZR  
    #person: 声音 (0: 女, 1: 男, 3: 特殊声音, 4: 特殊声音, 缺省0)  
    person: 4  
    #speed: 语速0-9 (缺省5)  
    speed: 5  
    #pitch: 语调0-9 (缺省5)  
    pitch: 5  
    #volume: 音量0-15 (缺省5)  
    volume: 15
```

## 5.人脸注册

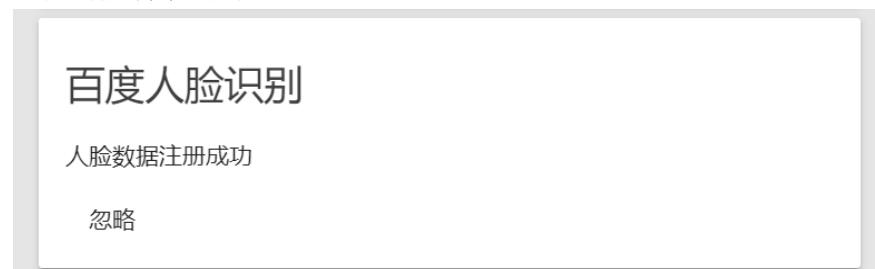
- 将照片传到HomeAssistant目录下的uploadpics目录下
- 选择注册人脸服务进行注册



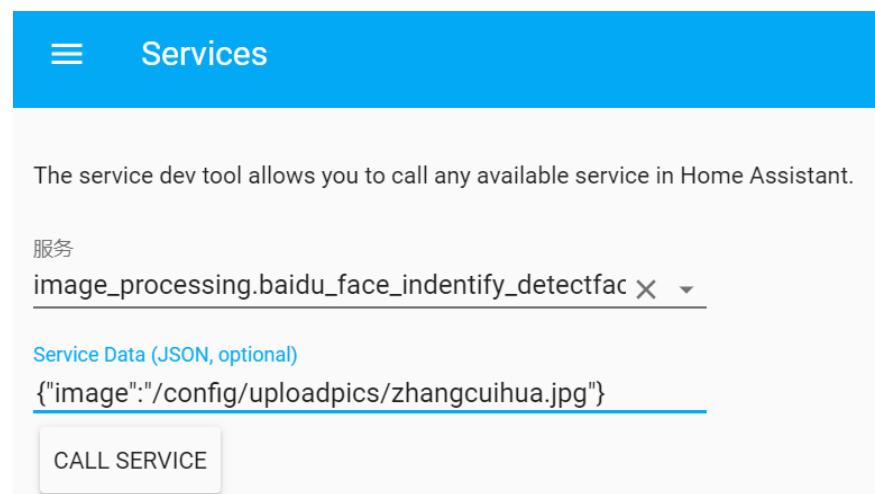
依照此格式填写： { "user\_info": "乔碧  
萝", "image": "/config/uploadpics/zhangcuihua.jpg", "uid": "tank" }

**user\_info**为用户标识，识别出人脸时候系统会显示这个名称 **uid**用于查找删除人脸数据 **image**为上传照片的路径

注册成功后会弹出提示



- 可对注册的人脸进行颜值检测



依照此格式填写： { "image": "/config/uploadpics/zhangcuihua.jpg" }

检测成功会返回结果

## 百度人脸识别

-唯一标识:9742383aa78d772ab4f7b187a499133b  
-人脸位置-离左边界距离:187.92  
-人脸位置-离上边界距离:3.6  
-人脸位置-宽度:257  
-人脸位置-高度:242  
-人脸位置-人脸框相对于竖直方向的顺时针旋转角[-180(逆时针),180(顺时针)]:0  
-人脸置信度[0最小、1最大]:0.88  
-三维旋转之左右旋转角[-90(左),90(右)]:-17.56  
-三维旋转之俯仰角度[-90(上),90(下)]:10.27  
-平面内旋转角[-180(逆时针),180(顺时针)]:-0.94  
-年龄:25  
-美丑打分[范围0-100, 越大表示越美]:43.26  
-表情-类型:微笑  
-表情-置信度[0-1]:0.98  
-脸型-类型:椭圆  
-脸型-置信度[0-1]:0.52  
-性别-类型:女性  
-性别-置信度[0-1]:0.99  
-是否带眼镜-类型:无  
-是否带眼镜-置信度[0-1]:1  
-人种-类型:黄种人  
-人种-置信度[0-1]:1  
-人脸质量信息-遮挡概率-左眼遮挡[0-1]:0  
-人脸质量信息-遮挡概率-右眼遮挡[0-1]:0  
-人脸质量信息-遮挡概率-鼻子遮挡[0-1]:0  
-人脸质量信息-遮挡概率-嘴巴遮挡[0-1]:0  
-人脸质量信息-遮挡概率-左脸颊遮挡[0-1]:0.08  
-人脸质量信息-遮挡概率-右脸颊遮挡[0-1]:0  
-人脸质量信息-遮挡概率:0  
-人脸质量信息-人脸模糊程度(0清晰, 1模糊):0  
-人脸质量信息-光照程度(0-255):149  
-人脸质量信息-人脸完整度(0-1):1

## 6. 自动化配置 **automations.yaml**

```
....  
....  
....  
  
# 当检测到人脸（人脸识别结果大于0），就执行tts和开关操作。  
- id: baiduface  
  alias: face_indentify  
  trigger:  
    - entity_id: sensor.ren_lian_shi_bie  
      platform: state  
      to: 'True'  
  action:  
    - data_template:  
        entity_id: media_player.kodi  
        message: >  
  
        service: tts.baidu_say  
        service: switch.turn_on  
        data:  
          - entity_id: switch.door
```

## 执行人脸解锁

- 将安卓手机放置在大门边上
- 可以将另一个大的显示器放在边上，全屏实时显示手机拍摄的实时画面
- 当人靠近手机摄像头的时候，如果能够正确识别，则会播放语音问候，并执行红外线开关操作

在这个基础上，可以扩展出各种其他的应用，基本过程就是通过正确的人脸识别，触发其他物联网操作。比如在教室门口识别到教师后，就自动执行打开投影仪、放下投影幕布，拉上窗帘等一系列上课操作。

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook 修订时间：2020-02-26

# 资料下载

相关资源, 请[点击这里](#)来下载 账号: sli 密码: sli

© 北京师范大学智慧学习研究院 all right reserved, powered by Gitbook修订时间: 2020-03-13