# 人工智能课程文档-课程活动部分

<font size=5>目录</font>

<!-- @import "[TOC]" {cmd="toc" depthFrom=2 depthTo=6 orderedList=false }-->

<!-- code\_chunk\_output -->

\* [\*\*课程概述\*\*](#课程概述)

\* [课程简介](#课程简介)

\* [课程目标](#课程目标)

\* [课程大纲](#课程大纲)

\* [预备要求](#预备要求)

\* [学分授予](#学分授予)

\* [章节测试成绩](#章节测试成绩)

\* [作业成绩](#作业成绩)

\* [交流讨论成绩](#交流讨论成绩)

\* [期末考试成绩](#期末考试成绩)

\* [参考资源](#参考资源)

\* [\*\*前导\*\*](#前导)

\* [计算机概述](#计算机概述)

\* [计算机的发展](#计算机的发展)

\* [计算机的操作系统](#计算机的操作系统)

\* [计算机语言与程序设计](#计算机语言与程序设计)

\* [人工智能概述](#人工智能概述)

\* [人工智能的发展](#人工智能的发展)

\* [人工智能的关键技术](#人工智能的关键技术)

\* [云计算与大数据](#云计算与大数据)

\* [知识图谱](#知识图谱)

\* [机器学习](#机器学习)

\* [自然语言处理](#自然语言处理)

\* [计算机视觉](#计算机视觉)

\* [智能机器人](#智能机器人)

\* [类脑智能](#类脑智能)

\* [人工智能的应用](#人工智能的应用)

\* [人工智能在金融领域的应用](#人工智能在金融领域的应用)

\* [人工智能在医疗领域的应用](#人工智能在医疗领域的应用)

\* [人工智能在家居领域的应用](#人工智能在家居领域的应用)

\* [人工智能在汽车领域的应用](#人工智能在汽车领域的应用)

\* [人工智能在零售领域的应用](#人工智能在零售领域的应用)

\* [人工智能在教育领域的应用](#人工智能在教育领域的应用)

\* [物联网概述](#物联网概述)

\* [物联网的发展](#物联网的发展)

\* [物联网的关键技术](#物联网的关键技术)

\* [边缘计算](#边缘计算)

\* [物联网的应用](#物联网的应用)

\* [物联网在家居领域的应用](#物联网在家居领域的应用)

\* [物联网在教育领域的应用](#物联网在教育领域的应用)

\* [\*\*环境准备\*\*](#环境准备)

\* [笔记本电脑](#笔记本电脑)

\* [Anaconda](#anaconda)

\* [基本命令概览](#基本命令概览)

\* [创建环境](#创建环境)

\* [进入环境](#进入环境)

\* [安装指定包](#安装指定包)

\* [其他命令](#其他命令)

\* [setup](#setup)

\* [VSCode](#vscode)

\* [setup](#setup-1)

\* [CP2102驱动](#cp2102驱动)

\* [Arduino IDE](#arduino-ide)

\* [setup](#setup-2)

\* [下载课程所需文件](#下载课程所需文件)

\* [NVIDIA Jetson Nano](#nvidia-jetson-nano)

\* [树莓派](#树莓派)

\* [Arduino](#arduino)

\* [ESP8266](#esp8266)

\* [传感器](#传感器)

\* [其他](#其他)

\* [\*\*第一章 物联网与传感器\*\*](#第一章-物联网与传感器)

\* [第一节 使用esp8266在网页上读取传感器数据，绘制实时变化曲线（待调整）](#第一节-使用esp8266在网页上读取传感器数据绘制实时变化曲线待调整)

\* [\*\*学习流程\*\*](#学习流程)

\* [课程引入 （15分钟）](#课程引入-15分钟)

\* [基本任务 （30分钟）](#基本任务-30分钟)

\* [活动总结 （10分钟）](#活动总结-10分钟)

\* [\*\*活动目标（PPT）\*\*](#活动目标ppt)

\* [\*\*背景知识（PPT）\*\*](#背景知识ppt)

\* [物联网背景知识](#物联网背景知识)

\* [如何使用开发板读取传感器数据](#如何使用开发板读取传感器数据)

\* [\*\*硬件准备（PPT）\*\*](#硬件准备ppt)

\* [硬件清单](#硬件清单)

\* [硬件连接](#硬件连接)

\* [\*\*程序及操作（PPT）\*\*](#程序及操作ppt)

\* [操作步骤-简单读取](#操作步骤-简单读取)

\* [操作步骤-绘制实时变化曲线](#操作步骤-绘制实时变化曲线)

\* [代码详解](#代码详解)

\* [第二节 WiFi遥控小车](#第二节-wifi遥控小车)

\* [\*\*demo\*\*](#demo)

\* [\*\*学习流程\*\*](#学习流程-1)

\* [课程引入 （15分钟）](#课程引入-15分钟-1)

\* [基本任务 （35分钟）](#基本任务-35分钟)

\* [活动总结 （10分钟）](#活动总结-10分钟-1)

\* [\*\*活动目标\*\*](#活动目标)

\* [\*\*背景知识\*\*](#背景知识)

\* [物联网背景知识](#物联网背景知识-1)

\* [开发板背景知识](#开发板背景知识)

\* [\*\*硬件准备\*\*](#硬件准备)

\* [硬件清单](#硬件清单-1)

\* [硬件连接](#硬件连接-1)

\* [\*\*程序及操作\*\*](#程序及操作)

\* [操作步骤](#操作步骤)

\* [代码详解](#代码详解-1)

\* [Part 1.3 WiFi机械臂](#part-13-wifi机械臂)

\* [\*\*demo\*\*](#demo-1)

\* [\*\*学习流程\*\*](#学习流程-2)

\* [课程引入 （15分钟）](#课程引入-15分钟-2)

\* [基本任务 （35分钟）](#基本任务-35分钟-1)

\* [活动总结 （10分钟）](#活动总结-10分钟-2)

\* [\*\*活动目标\*\*](#活动目标-1)

\* [\*\*背景知识\*\*](#背景知识-1)

\* [舵机](#舵机)

\* [舵机的工作原理](#舵机的工作原理)

\* [脉冲宽度调制（PWM）](#脉冲宽度调制pwm)

\* [占空比](#占空比)

\* [\*\*硬件准备\*\*](#硬件准备-1)

\* [硬件清单](#硬件清单-2)

\* [硬件连接](#硬件连接-2)

\* [\*\*程序及操作\*\*](#程序及操作-1)

\* [操作步骤](#操作步骤-1)

\* [代码详解](#代码详解-2)

\* [Part 1.4 esp32网络摄像头与人脸识别](#part-14-esp32网络摄像头与人脸识别)

\* [\*\*学习流程\*\*](#学习流程-3)

\* [课程引入 （15分钟）](#课程引入-15分钟-3)

\* [基本任务 （35分钟）](#基本任务-35分钟-2)

\* [活动总结 （10分钟）](#活动总结-10分钟-3)

\* [\*\*活动目标\*\*](#活动目标-2)

\* [\*\*背景知识\*\*](#背景知识-2)

\* [A](#a)

\* [B](#b)

\* [\*\*硬件准备\*\*](#硬件准备-2)

\* [硬件清单](#硬件清单-3)

\* [硬件连接](#硬件连接-3)

\* [注意：`IO0`口需要和它边上的`GND`口用一根杜邦线连接到一起，这样才可以正常上传代码](#注意io0口需要和它边上的gnd口用一根杜邦线连接到一起这样才可以正常上传代码)

\* [\*\*程序及操作\*\*](#程序及操作-2)

\* [操作步骤](#操作步骤-2)

\* [代码详解](#代码详解-3)

\* [Part 1.5 综合与进阶](#part-15-综合与进阶)

\* [\*\*参考\*\*](#参考)

\* [\*\*demo\*\*](#demo-2)

\* [\*\*抓取大赛\*\*](#抓取大赛)

\* [比赛规则](#比赛规则)

\* [尾声](#尾声)

\* [\*\*Part 2 物联网开源平台Home Assistant创意应用\*\*](#part-2-物联网开源平台home-assistant创意应用)

\* [Part 2.1 Home Assistant安装和配置](#part-21-home-assistant安装和配置)

\* [\*\*硬件准备\*\*](#硬件准备-3)

\* [硬件清单](#硬件清单-4)

\* [硬件连接](#硬件连接-4)

\* [\*\*程序及操作\*\*](#程序及操作-3)

\* [安装过程](#安装过程)

\* [Part 2.2 Home Assistant控制esp8266彩色灯](#part-22-home-assistant控制esp8266彩色灯)

\* [\*\*硬件准备\*\*](#硬件准备-4)

\* [\*\*程序及操作\*\*](#程序及操作-4)

\* [Part 2.3 Home Assistant人脸解锁](#part-23-home-assistant人脸解锁)

\* [\*\*硬件准备\*\*](#硬件准备-5)

\* [\*\*程序及操作\*\*](#程序及操作-5)

\* [\*\*Part 3 进阶项目-物联网机器人小绿\*\*](#part-3-进阶项目-物联网机器人小绿)

\* [Part 3.1 组装一个小绿机器人](#part-31-组装一个小绿机器人)

\* [\*\*硬件准备\*\*](#硬件准备-6)

\* [\*\*程序及操作\*\*](#程序及操作-6)

\* [Part 3.2 训练语音识别，开始和小绿聊天](#part-32-训练语音识别开始和小绿聊天)

\* [\*\*硬件准备\*\*](#硬件准备-7)

\* [\*\*程序及操作\*\*](#程序及操作-7)

\* [Part 3.3 让小绿听你指挥，控制一切](#part-33-让小绿听你指挥控制一切)

\* [\*\*硬件准备\*\*](#硬件准备-8)

\* [\*\*程序及操作\*\*](#程序及操作-8)

\* [Part 3.4 使用Google Blockly来控制小绿](#part-34-使用google-blockly来控制小绿)

\* [\*\*硬件准备\*\*](#硬件准备-9)

\* [\*\*程序及操作\*\*](#程序及操作-9)

\* [Part 3.5 使用OpenPose让小绿实时模仿你的动作](#part-35-使用openpose让小绿实时模仿你的动作)

\* [\*\*硬件准备\*\*](#硬件准备-10)

\* [\*\*程序及操作\*\*](#程序及操作-10)

\* [\*\*Chapter 2 人工智能与机器人\*\*(jetson | 树莓派+PC)](#chapter-2-人工智能与机器人jetson-树莓派pc)

\* [\*\*Part 1 人工智能算法相关案例体验\*\*](#part-1-人工智能算法相关案例体验)

\* [Part 1.1 Tensorflow训练自定义图片分类器](#part-11-tensorflow训练自定义图片分类器)

\* [\*\*硬件准备\*\*](#硬件准备-11)

\* [\*\*程序及操作\*\*](#程序及操作-11)

\* [Part 1.2 使用RNN来生成古诗词](#part-12-使用rnn来生成古诗词)

\* [\*\*硬件准备\*\*](#硬件准备-12)

\* [\*\*程序及操作\*\*](#程序及操作-12)

\* [Part 1.3 训练一个简单的游戏AI（Deep Q Network）](#part-13-训练一个简单的游戏aideep-q-network)

\* [\*\*硬件准备\*\*](#硬件准备-13)

\* [\*\*程序及操作\*\*](#程序及操作-13)

\* [Part 1.4 使用进化算法来训练超级马里奥](#part-14-使用进化算法来训练超级马里奥)

\* [\*\*硬件准备\*\*](#硬件准备-14)

\* [\*\*程序及操作\*\*](#程序及操作-14)

\* [Part 1.5 基于opencv在raspberry 上的人脸识别操作](#part-15-基于opencv在raspberry-上的人脸识别操作)

\* [\*\*硬件准备\*\*](#硬件准备-15)

\* [\*\*环境准备\*\*](#环境准备-1)

\* [\*\*程序及操作\*\*](#程序及操作-15)

\* [Part 1.6 Neural Style 图像风格迁移](#part-16-neural-style-图像风格迁移)

\* [\*\*准备\*\*](#准备)

\* [VGG 网络](#vgg-网络)

\* [环境准备](#环境准备-2)

\* [\*\*程序及操作\*\*](#程序及操作-16)

\* [\*\*Part 2 自动避障小车\*\*](#part-2-自动避障小车)

\* [Part 2.1 环境准备与硬件搭建](#part-21-环境准备与硬件搭建)

\* [\*\*硬件准备\*\*](#硬件准备-16)

\* [\*\*程序及操作\*\*](#程序及操作-17)

\* [Part 2.2 通过超声波传感器进行避障](#part-22-通过超声波传感器进行避障)

\* [\*\*硬件准备\*\*](#硬件准备-17)

\* [\*\*程序及操作\*\*](#程序及操作-18)

\* [\*\*Part 3 自动追踪小车\*\*](#part-3-自动追踪小车)

\* [Part 3.1 环境准备与硬件搭建](#part-31-环境准备与硬件搭建)

\* [\*\*硬件准备\*\*](#硬件准备-18)

\* [\*\*程序及操作\*\*](#程序及操作-19)

\* [Part 3.2 OpenCV机械臂自动抓取特定形状物体](#part-32-opencv机械臂自动抓取特定形状物体)

\* [\*\*硬件准备\*\*](#硬件准备-19)

\* [\*\*程序及操作\*\*](#程序及操作-20)

\* [Part 3.3 OpenCV分类器训练，让小车追踪特定物体](#part-33-opencv分类器训练让小车追踪特定物体)

\* [\*\*硬件准备\*\*](#硬件准备-20)

\* [\*\*程序及操作\*\*](#程序及操作-21)

\* [\*\*Part 4 无人驾驶小车\*\*](#part-4-无人驾驶小车)

\* [Part 4.1 环境准备与硬件搭建](#part-41-环境准备与硬件搭建)

\* [\*\*硬件准备\*\*](#硬件准备-21)

\* [硬件清单](#硬件清单-5)

\* [硬件连接](#硬件连接-5)

\* [\*\*程序及操作\*\*](#程序及操作-22)

\* [操作步骤](#操作步骤-3)

\* [Part 4.2 电机和摄像头驱动测试](#part-42-电机和摄像头驱动测试)

\* [\*\*程序及操作\*\*](#程序及操作-23)

\* [操作步骤-电机测试](#操作步骤-电机测试)

\* [操作步骤-摄像头测试](#操作步骤-摄像头测试)

\* [Part 4.3 无人驾驶数据采集及训练](#part-43-无人驾驶数据采集及训练)

\* [\*\*硬件准备\*\*](#硬件准备-22)

\* [硬件清单](#硬件清单-6)

\* [硬件搭建-跑道](#硬件搭建-跑道)

\* [\*\*程序及操作\*\*](#程序及操作-24)

\* [操作步骤-驾驶数据采集](#操作步骤-驾驶数据采集)

\* [操作步骤-驾驶数据训练](#操作步骤-驾驶数据训练)

\* [Part 4.4 开始无人驾驶](#part-44-开始无人驾驶)

\* [\*\*程序及操作\*\*](#程序及操作-25)

\* [操作步骤](#操作步骤-4)

<!-- /code\_chunk\_output -->

---

## \*\*课程概述\*\*

本课程……

### 课程简介

### 课程目标

### 课程大纲

### 预备要求

### 学分授予

##### 章节测试成绩

##### 作业成绩

##### 交流讨论成绩

##### 期末考试成绩

### 参考资源

---

## \*\*前导\*\*

本章内容……

### 计算机概述

##### 计算机的发展

##### 计算机的操作系统

##### 计算机语言与程序设计

### 人工智能概述

##### 人工智能的发展

##### 人工智能的关键技术

###### 云计算与大数据

###### 知识图谱

###### 机器学习

###### 自然语言处理

###### 计算机视觉

###### 智能机器人

###### 类脑智能

##### 人工智能的应用

###### 人工智能在金融领域的应用

###### 人工智能在医疗领域的应用

###### 人工智能在家居领域的应用

###### 人工智能在汽车领域的应用

###### 人工智能在零售领域的应用

###### 人工智能在教育领域的应用

### 物联网概述

##### 物联网的发展

##### 物联网的关键技术

###### 边缘计算

##### 物联网的应用

###### 物联网在家居领域的应用

###### 物联网在教育领域的应用

---

## \*\*环境准备\*\*

## 笔记本电脑

环境准备相关所有相关软件，\*\*请[点击这里](ftp://home.hass.live)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

### 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>

![Xnip2019-04-30\_10-37-45](https://md.hass.live/Xnip2019-04-30\_10-37-45.png?imageView2/0/interlace/1/q/46|imageslim)

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

```bash {.line-numbers}

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

```bash {.line-numbers}

conda activate 环境名字

//例如：

conda activate py27

```

##### 安装指定包

```bash {.line-numbers}

conda install 包名

//例如：安装OpenCV

conda install opencv

```

##### 其他命令

```bash {.line-numbers}

//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）

```bash {.line-numbers}

//创建一个名字为myenv的虚拟环境

conda create -n learn-ai

//激活myenv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 gi

```

### 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`

![2019-04-29 21.49.44](https://md.hass.live/2019-04-29%2021.49.44.gif)

3.输入`Shift`+`Alt`+`D`,输入GitHub Token和Gist Token([点击获取](http://hass.live:8999/))，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://dl.espressif.com/dl/package\_esp32\_index.json> 重启IDE

![arduino-config-1](https://md.hass.live/niji/2019-05-09-Xnip2019-05-09\_18-19-37.png)

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

![esp8266](https://md.hass.live/2019-04-29%2022.10.50.gif?imageView2/0/interlace/1/q/46|imageslim)

![esp32](https://md.hass.live/niji/2019-05-09-2019-05-09%2018.22.25.gif?imageView2/0/interlace/1/q/46|imageslim)

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

![dht](https://md.hass.live/niji/2019-05-22-Xnip2019-05-22\_13-50-38.png)

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

![ada2](https://md.hass.live/niji/2019-05-22-Xnip2019-05-22\_13-57-25.png)

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

![arduino-config-2](https://md.hass.live/arduinoconfig2.png?imageView2/0/interlace/1/q/46|imageslim)

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:

<center><img src=https://md.hass.live/niji/2019-05-07-Xnip2019-05-07\_23-41-38.png></center>

7.设置开发板和端口

<center><img src=https://md.hass.live/Xnip2019-05-05\_16-49-07.png?imageView2/0/interlace/1/q/46|imageslim></center>

### 下载课程所需文件

Windows用户执行`git clone https://github.com/nijisakai/learn-ai.git C:/learn-ai`

文件被下载到C盘根目录下面的learn-ai文件夹

macOS用户打开`终端`

Windows用户打开`Anaconda Prompt`

macOS用户执行`git clone https://github.com/nijisakai/learn-ai.git ~/Desktop/learn-ai/`

文件被下载到桌面下面的learn-ai文件夹

## NVIDIA Jetson Nano

## 树莓派

## Arduino

## ESP8266

esp8266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

本章内容是关于使用可编程的开源硬件，将功能点进行分解，并最终实现综合项目

## 传感器

## 其他

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://192.168.0.1>或默认地址

---

## \*\*第一章 物联网与传感器\*\*

物联网给人十分大方哥哥啊

### 第一节 使用esp8266在网页上读取传感器数据，绘制实时变化曲线（待调整）

这部分让你熟悉操作esp8266的步骤。是第一章的基础

包括功能提出和实现，硬件连接，上传的参数调节和html文件在本地服务器中的打开，传感器数据的实时呈现等，并使用Chart.js来绘制实时变化曲线

这部分主要包括两种传感器的读取，为温湿度传感器和超声波传感器

#### \*\*学习流程\*\*

##### 课程引入 （15分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[活动目标](#活动目标) | 了解物联网基本概念 | 5分钟

[背景知识](#背景知识) | 熟悉实验中涉及到的软硬件 | 10分钟

</center>

##### 基本任务 （30分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[硬件准备](#硬件准备) | 将硬件按文档进行连接 | 5分钟

[程序及操作](#程序及操作) | 完成程序及操作文档部分 | 25分钟

</center>

##### 活动总结 （10分钟）

#### \*\*活动目标（PPT）\*\*

\* 了解物联网的基本概念

\* 了解使用开发板读取传感器的基本原理

\* 熟悉使用Arduino IDE烧录固件的操作流程

#### \*\*背景知识（PPT）\*\*

##### 物联网背景知识

<center><iframe src="https://player.bilibili.com/player.html?aid=46814591&cid=82000363&page=1" width="800" height="600" scrolling="no" border="0" frameborder="no" framespacing="0" allowfullscreen="true"> </iframe></center>

##### 如何使用开发板读取传感器数据

esp8266是WiFi串口模块，功能简单来讲就是：从WiFi接收到数据，串口输出；从串口接收数据，WiFi输出数据。

通过自带的GPIO口连接传感器，传感器将环境数据转化为电信号发送给esp8266读取、处理并输出。

#### \*\*硬件准备（PPT）\*\*

##### 硬件清单

\* esp8266主板

\* 温湿度传感器（型号为DHT11或DHT22）

\* 超声波传感器（型号为HC-SR04）

\* 杜邦线、数据线

##### 硬件连接

<center><img src=https://md.hass.live/Xnip2019-05-05\_11-52-46.png?imageView2/0/interlace/1/q/46|imageslim></center>

图解

<center><img src=https://md.hass.live/Xnip2019-05-05\_12-02-00.png?imageView2/0/interlace/1/q/46|imageslim></center>

图解

#### \*\*程序及操作（PPT）\*\*

##### 操作步骤-简单读取

1.打开`learn-ai`文件夹，打开路径`chapter1/part1/esp8266\_projects/esp8266\_dht11\_http`

2.将esp8266通过数据线连接到电脑

3.使用Arduino IDE打开文件`esp8266\_dht11\_https.ino`

4.记得把前面的[环境准备](#setup-2)部分再次确认，将环境正确配置，然后点击上传按钮进行上传

<center><img src=https://md.hass.live/niji/2019-05-08-Xnip2019-05-08\_10-15-02.png?imageView2/0/interlace/1/q/46|imageslim></center>

5.打开[路由器管理地址](http://192.168.0.1)，esp8266此时应该已经加入到了局域网中，查看esp8266获取到的路由器地址

6.在浏览器中打开esp8266获取到的局域网地址，查看温湿度传感器的读数

7.连接另一个esp8266开发板，打开路径`chapter1/part1/esp8266\_projects/esp8266\_ultrasonic\_http`,再次执行2-6步骤来使用超声波传感器

##### 操作步骤-绘制实时变化曲线

1.打开`learn-ai`文件夹，打开路径`chapter1/part1/esp8266\_projects/esp8266\_dht11\_http\_chartjs`

2.将esp8266通过数据线连接到电脑

3.使用Arduino IDE打开文件 `esp8266\_dht11\_http\_chartjs.ino`

4.记得把前面的[环境准备](#setup-2)部分再次确认，将环境正确配置，然后点击上传按钮进行上传

<center><img src='https://md.hass.live/niji/2019-05-08-Xnip2019-05-08\_10-15-02.png?imageView2/0/interlace/1/q/46|imageslim'></center>

5.打开[路由器管理地址](http://192.168.0.1)，esp8266此时应该已经加入到了局域网中，查看esp8266获取到的路由器地址

6.在浏览器中打开esp8266获取到的局域网地址，查看温湿度传感器的读数

<center><img src='https://md.hass.live/niji/2019-05-08-Xnip2019-05-08\_10-13-13.png?imageView2/0/interlace/1/q/46|imageslim'></center>

##### 代码详解

\* 温湿度传感器

```arduino {.line-numbers} {cmd}

//两个反斜杠代表单行注释，这后面的文字不会被执行

/\*

被星号和反斜杠包围的部分可以进行多行注释

\*/

/\*使用多行注释符

Arduino程序包含了两个主要的方法，分别是

void setup()和void loop()

方法后面写一对{},在这里面写方法的具体内容。看起来像是这样：

void setup(){

自己规定的语句

自己规定的语句

自己规定的语句

}

void loop(){

自己规定的语句

自己规定的语句

自己规定的语句

}

\*/

//使用 #include来引用一些附加的功能。

#include <esp8266WiFi.h>

#include <esp8266WebServer.h>

#include "DHT.h"

/////////////////////////////////////////////

////////////////下面的部分请你配置//////////////

/////////////////////////////////////////////

// 1️⃣选择使用哪一种DHT传感器。通过去掉前面的注释符来使语句生效

#define DHTTYPE DHT11 // DHT 11

//#define DHTTYPE DHT21 // DHT 21 (AM2301)

//#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321

// 2️⃣把引号里的内容替换为自己路由器的用户名和密码

const char\* ssid = "AI"; // 输入路由器用户名

const char\* password = "raspberry"; //输入路由器密码

// 3️⃣把DHT传感器的Data接口连接到D8。此处与硬件连线对应即可

uint8\_t DHTPin = D8;

// 4️⃣默认为80端口，如果改为其他端口，访问的时候在最后加上（:端口名）

esp8266WebServer server(80);

/////////////////////////////////////////////

////////////////上面的部分请你配置//////////////

/////////////////////////////////////////////

// 初始化你连接的DHT传感器

DHT dht(DHTPin, DHTTYPE);

//定义两个float变量来读取温湿度

float Temperature; //温度

float Humidity; //湿度

//开始最后的准备工作

void setup() {

Serial.begin(115200);

delay(100);

pinMode(DHTPin, INPUT);

dht.begin();

Serial.println("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

//检查连接是否正常

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected..!");

Serial.print("Got IP: "); Serial.println(WiFi.localIP());

server.on("/", handle\_OnConnect);

server.onNotFound(handle\_NotFound);

server.begin();

Serial.println("http server started");

}

void handle\_OnConnect() {

Temperature = dht.readTemperature();

Humidity = dht.readHumidity();

server.send(200, "text/html", SendHTML(Temperature,Humidity));

}

void handle\_NotFound(){

server.send(404, "text/plain", "Not found");

}

String SendHTML(float Temperaturestat,float Humiditystat){

String ptr = "<!DOCTYPE html> <html>\n";

ptr +="<head><meta charset=\"UTF-8\"><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">\n";

ptr +="<title>esp8266 DHT11</title>\n";

ptr +="<script>\n";

ptr +="setInterval(loadDoc,200);\n";

ptr +="function loadDoc() {\n";

ptr +="var xhttps = new XMLhttpsRequest();\n";

ptr +="xhttps.onreadystatechange = function() {\n";

ptr +="if (this.readyState == 4 && this.status == 200) {\n";

ptr +="document.getElementById(\"webpage\").innerHTML =this.responseText}\n";

ptr +="};\n";

ptr +="xhttps.open(\"GET\", \"/\", true);\n";

ptr +="xhttps.send();\n";

ptr +="}\n";

ptr +="</script>\n";

ptr +="</head>\n";

ptr +="<body>\n";

ptr +="<div id=\"webpage\">\n";

ptr +="<p>温度: ";

ptr +=(int)Temperaturestat;

ptr +="°C</p>";

ptr +="<p>湿度: ";

ptr +=(int)Humiditystat;

ptr +="%</p>";

ptr +="</div>\n";

ptr +="</body>\n";

ptr +="</html>\n";

return ptr;

}

//所有准备工作就绪，开始工作

void loop() {

server.handleClient();

}

```

\* 超声波传感器

```arduino {.line-numbers}

#include <Arduino.h>

#include <esp8266WiFi.h>

#include <esp8266WiFiMulti.h>

#include <esp8266httpsClient.h>

#include <esp8266WebServer.h>

// utlrasonic pinout

#define ULTRASONIC\_TRIG\_PIN 5 // pin TRIG to D1

#define ULTRASONIC\_ECHO\_PIN 4 // pin ECHO to D2

const char\* WiFi\_ssid = "AI"; // SSID

const char\* WiFi\_password = "raspberry"; // WIFI

esp8266WebServer server(80);

esp8266WiFiMulti WiFiMulti;

void setup() {

Serial.begin(115200);

Serial.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Serial.println("\*\*\*\*\*\*\*\*\*\* Program Start : Connect Ultrasonic HC-SR04 + esp8266 to AskSensors over http");

Serial.println("Wait for WiFi... ");

Serial.print("\*\*\*\*\*\*\*\*\*\* connecting to WIFI : ");

Serial.println(WiFi\_ssid);

WiFi.begin(WiFi\_ssid, WiFi\_password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("-> WiFi connected");

Serial.println("-> IP address: ");

Serial.println(WiFi.localIP());

// ultraonic setup

pinMode(ULTRASONIC\_TRIG\_PIN, OUTPUT);

pinMode(ULTRASONIC\_ECHO\_PIN, INPUT);

server.on("/", handle\_OnConnect);

server.onNotFound(handle\_NotFound);

server.begin();

}

void handle\_OnConnect() {

long duration, distance;

digitalWrite(ULTRASONIC\_TRIG\_PIN, LOW);

delayMicroseconds(2);

digitalWrite(ULTRASONIC\_TRIG\_PIN, HIGH);

delayMicroseconds(10);

digitalWrite(ULTRASONIC\_TRIG\_PIN, LOW);

duration = pulseIn(ULTRASONIC\_ECHO\_PIN, HIGH);

distance = (duration/2) / 29.1;

server.send(200, "text/html", SendHTML(distance));

}

void handle\_NotFound(){

server.send(404, "text/plain", "Not found");

}

String SendHTML(long distance){

String ptr = "<!DOCTYPE html> <html>\n";

ptr +="<head><meta charset=\"UTF-8\"><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">\n";

ptr +="<title>esp8266 DHT11</title>\n";

ptr +="<script>\n";

ptr +="setInterval(loadDoc,200);\n";

ptr +="function loadDoc() {\n";

ptr +="var xhttps = new XMLhttpsRequest();\n";

ptr +="xhttps.onreadystatechange = function() {\n";

ptr +="if (this.readyState == 4 && this.status == 200) {\n";

ptr +="document.getElementById(\"webpage\").innerHTML =this.responseText}\n";

ptr +="};\n";

ptr +="xhttps.open(\"GET\", \"/\", true);\n";

ptr +="xhttps.send();\n";

ptr +="}\n";

ptr +="</script>\n";

ptr +="</head>\n";

ptr +="<body>\n";

ptr +="<div id=\"webpage\">\n";

ptr +="<p>距离: ";

ptr +=(float)distance;

ptr +="厘米</p>";

ptr +="</div>\n";

ptr +="</body>\n";

ptr +="</html>\n";

return ptr;

}

void loop() {

server.handleClient();

}

```

\* 温湿度传感器变化曲线

```arduino {.line-numbers}

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

#include "index.h" //Our HTML webpage contents with javascripts

#include "DHTesp.h" //DHT11 Library for ESP

#define LED 2 //On board LED

#define DHTpin 15 //D8 of NodeMCU is GPIO15

DHTesp dht;

const char\* ssid = "AI";

const char\* password = "raspberry";

ESP8266WebServer server(80); //Server on port 80

void handleRoot() {

String s = MAIN\_page; //Read HTML contents

server.send(200, "text/html", s); //Send web page

}

float humidity, temperature;

void handleADC() {

int a = analogRead(A0);

String data = "{\"ADC\":\""+String(a)+"\", \"Temperature\":\""+ String(temperature) +"\", \"Humidity\":\""+ String(humidity) +"\"}";

digitalWrite(LED,!digitalRead(LED)); //Toggle LED on data request ajax

server.send(200, "text/plane", data); //Send ADC value, temperature and humidity JSON to client ajax request

//Get Humidity temperatue data after request is complete

//Give enough time to handle client to avoid problems

delay(dht.getMinimumSamplingPeriod());

humidity = dht.getHumidity();

temperature = dht.getTemperature();

Serial.print(humidity, 1);

Serial.print(temperature, 1);

Serial.print(dht.toFahrenheit(temperature), 1);

}

//==============================================================

// SETUP

//==============================================================

void setup()

{

Serial.begin(115200);

Serial.println();

dht.setup(DHTpin, DHTesp::DHT11); //for DHT11 Connect DHT sensor to GPIO 17

//dht.setup(DHTpin, DHTesp::DHT22); //for DHT22 Connect DHT sensor to GPIO 17

WiFi.begin(ssid, password); //Connect to your WiFi router

Serial.println("");

//Onboard LED port Direction output

pinMode(LED,OUTPUT);

// Wait for connection

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

//If connection successful show IP address in serial monitor

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("IP address: ");

Serial.println(WiFi.localIP()); //IP address assigned to your ESP

server.on("/", handleRoot); //Which routine to handle at root location. This is display page

server.on("/readADC", handleADC); //This page is called by java Script AJAX

server.begin(); //Start server

Serial.println("HTTP server started");

}

//==============================================================

// LOOP

//==============================================================

void loop()

{

server.handleClient(); //Handle client requests

}

```

### 第二节 WiFi遥控小车

使用esp8266，通过网页端发送命令，遥控一辆小车，结合超声波传感器，实现自动避障

#### \*\*demo\*\*

<center><iframe src="http://hass.live:9000" width="800" height="300" scrolling="no" frameborder="0" mozallowfullscreen webkitallowfullscreen allowfullscreen></iframe></center>

#### \*\*学习流程\*\*

##### 课程引入 （15分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[活动目标](#活动目标-1) | 了解物联网基本概念 | 5分钟

[背景知识](#背景知识-1) | 熟悉实验中涉及到的软硬件 | 10分钟

</center>

##### 基本任务 （35分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[硬件准备](#硬件准备-1) | 将硬件按文档进行连接 | 5分钟

[程序及操作](#程序及操作-1) | 完成程序及操作文档部分 | 30分钟

</center>

##### 活动总结 （10分钟）

#### \*\*活动目标\*\*

\* 目标1:学会使用esp8266的服务器功能

\* 目标2:学会使用超声模块

\* 目标3:学会控制esp8266的gpio

#### \*\*背景知识\*\*

##### 物联网背景知识

<center><iframe src="https://player.bilibili.com/player.html?aid=46814591&cid=82000363&page=1" width="800" height="600" scrolling="no" border="0" frameborder="no" framespacing="0" allowfullscreen="true"> </iframe></center>

##### 开发板背景知识

文字描述

#### \*\*硬件准备\*\*

##### 硬件清单

\* esp8266主板

\* 电机扩展板 esp12E Motor Shield

\* 小车套件(3D打印的底盘和夹层，电机，车轮，铜柱等)

\* 杜邦线，数据线

##### 硬件连接

<center><img src=https://md.hass.live/niji/2019-05-07-Xnip2019-05-07\_17-10-07.png></center>

#### \*\*程序及操作\*\*

##### 操作步骤

1.打开`learn-ai`文件夹，打开路径`chapter1/part1/esp8266\_projects/esp8266\_WiFicar\_https`

2.将esp8266通过数据线连接到电脑

3.使用Arduino IDE打开文件`esp8266\_WiFicar\_https.ino`

4.记得把前面的[环境准备](#setup-2)部分再次确认，将环境正确配置，然后点击上传按钮进行上传

<center><img src='https://md.hass.live/niji/2019-05-08-Xnip2019-05-08\_10-15-02.png?imageView2/0/interlace/1/q/46|imageslim'></center>

5.点击`工具`菜单，选择`ESP8266 Sketch Data Upload`,会自动将项目目录下的data文件夹上传到esp8266开发板上

6.打开[路由器管理地址](http://192.168.0.1)，esp8266此时应该已经加入到了局域网中，查看esp8266获取到的路由器地址

7.将esp8266与电脑连接断开，连接到移动电源上

8.在浏览器中打开esp8266获取到的局域网地址，通过点击上下左右按钮或键盘的光标键来控制小车

##### 代码详解

\* WiFi小车程序

```arduino {.line-numbers}

#include <esp8266WiFi.h>

#include <WiFiClient.h>

#include <esp8266WebServer.h>

#include <esp8266mDNS.h>

#include <FS.h>

#define Motor\_AE D1 //Motor A/B,E enable,D Direction

#define Motor\_AD D3

#define Motor\_BE D2

#define Motor\_BD D4

#define R\_AHEAD HIGH

#define L\_AHEAD LOW

String command;

esp8266WebServer server(80);

const int led = 13;

void carInit(){

pinMode(Motor\_AE, OUTPUT);

pinMode(Motor\_AD, OUTPUT);

pinMode(Motor\_BE, OUTPUT);

pinMode(Motor\_BD, OUTPUT);

Serial.begin(115200);

Serial.println("Car begin");

}

void goAhead(){

digitalWrite(Motor\_AE, HIGH);

digitalWrite(Motor\_AD, L\_AHEAD);

digitalWrite(Motor\_BE, HIGH);

digitalWrite(Motor\_BD, R\_AHEAD);

}

void goBack(){

digitalWrite(Motor\_AE, HIGH);

digitalWrite(Motor\_AD, !L\_AHEAD);

digitalWrite(Motor\_BE, HIGH);

digitalWrite(Motor\_BD, !R\_AHEAD);

}

void goRight(){

digitalWrite(Motor\_BE, LOW);

digitalWrite(Motor\_AE, HIGH);

digitalWrite(Motor\_AD, L\_AHEAD);

}

void goLeft(){

digitalWrite(Motor\_BE, HIGH);

digitalWrite(Motor\_AE, LOW);

digitalWrite(Motor\_BD, R\_AHEAD);

}

void stopRobot(){

digitalWrite(Motor\_AE, LOW);

digitalWrite(Motor\_BE, LOW);

}

void handleNotFound(){

digitalWrite(led, 1);

String message = "File Not Found\n\n";

message += "URI: ";

message += server.uri();

message += "\nMethod: ";

message += (server.method() == https\_GET)?"GET":"POST";

message += "\nArguments: ";

message += server.args();

message += "\n";

for (uint8\_t i=0; i<server.args(); i++){

message += " " + server.argName(i) + ": " + server.arg(i) + "\n";

}

server.send(404, "text/plain", message);

digitalWrite(led, 0);

}

void setup(void){

carInit();

SPIFFS.begin();

uint8\_t mac[WL\_MAC\_ADDR\_LENGTH];

WiFi.softAPmacAddress(mac);

String macID = String(mac[WL\_MAC\_ADDR\_LENGTH - 2], HEX) + String(mac[WL\_MAC\_ADDR\_LENGTH - 1], HEX);

macID.toUpperCase();

String AP\_NameString = "Wifi Car - " + macID;

char AP\_NameChar[AP\_NameString.length() + 1];

memset(AP\_NameChar, 0, AP\_NameString.length() + 1);

for (int i = 0; i < AP\_NameString.length(); i++)

AP\_NameChar[i] = AP\_NameString.charAt(i);

const char\* ssid = "AI";

const char\* password = "raspberry";

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

Serial.println("");

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("IP address: ");

Serial.println(WiFi.localIP());

if (MDNS.begin("esp8266")) {

Serial.println("MDNS responder started");

}

//server.on("/", handleRoot);

server.serveStatic("/", SPIFFS, "/index.html");

server.on("/get", [](){

String uri = server.uri();

Serial.println(uri);

command = server.arg("command");

if(command == "forward")

goAhead();

else if(command == "backward")

goBack( );

else if(command == "left")

goLeft();

else if(command == "right")

goRight();

else if(command == "stop")

stopRobot();

Serial.println(command);

// setCorlor(red.toInt(),green.toInt(),blue.toInt());

server.send(200, "text/plain", String("set to ")+command);

});

server.onNotFound(handleNotFound);

server.begin();

Serial.println("http server started");

}

void loop(void){

server.handleClient();

}

```

### Part 1.3 WiFi机械臂

使用esp8266，通过网页端发送命令，控制多个舵机

#### \*\*demo\*\*

<center><iframe src="http://hass.live:9001" width="800" height="600" scrolling="yes" frameborder="0" mozallowfullscreen webkitallowfullscreen allowfullscreen></iframe></center>

#### \*\*学习流程\*\*

##### 课程引入 （15分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[活动目标](#活动目标-2) | 通过WiFi操控机械臂 | 5分钟

[背景知识](#背景知识-2) | 舵机的工作原理及脉冲宽度调制 | 10分钟

</center>

##### 基本任务 （35分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[硬件准备](#硬件准备-2) | 将硬件按文档进行连接 | 5分钟

[程序及操作](#程序及操作-2) | 完成程序及操作文档部分 | 30分钟

</center>

##### 活动总结 （10分钟）

#### \*\*活动目标\*\*

\* 目标1：了解舵机的原理和使用方法

\* 目标2：了解脉冲宽度调制（PWM）

\* 目标3：通过WiFi遥控机械臂

#### \*\*背景知识\*\*

##### 舵机

伺服电机通常被称为舵机，它是一种带有输出轴的小装置。当我们向伺服器发送一个控制信号时，输出轴就可以转到特定的位置。只要控制信号持续不变，伺服机构就会保持轴的角度位置不改变。如果控制信号发生变化，输出轴的位置也会相应发生变化。日常生活中，舵机常被用于遥控飞机、遥控汽车、机器人等。

##### 舵机的工作原理

<center>

![舵机结构](https://md.hass.live/niji/2019-05-27-%E8%88%B5%E6%9C%BA%E6%8B%86%E8%A7%A3.jpg)

</center>

舵机内部的控制电路、电位计（可变电阻器）和电机均被连接到电路板上。控制电路通过电位计可监控舵机的当前角度。

其工作流程为：控制信号 → 控制电路板 → 电机转动 → 齿轮组减速 → 舵盘转动 → 位置反馈电位计 → 控制电路板反馈。

<center>

![舵机转动图](https://md.hass.live/niji/2019-05-27-SERVO.gif)

</center>

##### 脉冲宽度调制（PWM）

脉冲宽度调制，英文名Pulse Width Modulation，缩写为PWM，它是通过对一系列脉冲的宽度进行调制，等效出所需要的波形，对模拟信号电平进行数字编码。

##### 占空比

占空比是指在一个周期内，信号处于高电平的时间占据整个信号周期的百分比。

<center>

![占空比](https://md.hass.live/niji/2019-05-27-Xnip2019-05-27\_09-37-35.png)

</center>

上图信号一个周期的时间为4ms，其中高电平时间为1ms。占空比为：$$ \frac {高电平时间} {一个周期} = \frac {1ms} {4ms} = 25\% $$

#### \*\*硬件准备\*\*

##### 硬件清单

\* esp8266主板

\* 电机扩展板 esp12E Motor Shield

\* 舵机

\* 杜邦线、数据线

\* 机械臂和零件

##### 硬件连接

<center><img src=https://md.hass.live/Xnip2019-05-05\_11-57-01.png?imageView2/0/interlace/1/q/46|imageslim></center>

<center>此处表示舵机连接到了D0口，最多可以连9个舵机（D0-D9）<br>

<font color=orange>黄色</font>-信号D（DATA）<br>

<font color=red>红色</font>-正极V（VCC）<br>

<font color=brown>棕色</font>-负极G（GND）<br>

</center>

#### \*\*程序及操作\*\*

##### 操作步骤

1. 打开`learn-ai`文件夹，打开路径`chapter1/part1/esp8266\_projects/esp8266\_servoarm\_http`

2. 将esp8266通过数据线连接到电脑

3. 使用Arduino IDE打开文件`esp8266\_servoarm\_http.ino`

4. 记得把前面的[环境准备](#setup-2)部分再次确认，将环境正确配置，然后点击上传按钮进行上传

<center><img src='https://md.hass.live/niji/2019-05-08-Xnip2019-05-08\_10-15-02.png?imageView2/0/interlace/1/q/46|imageslim'>

</center>

5. 点击`工具`菜单，选择`ESP8266 Sketch Data Upload`,会自动将项目目录下的data文件夹上传到esp8266开发板上

6. 打开[路由器管理地址](http://192.168.0.1)，esp8266此时应该已经加入到了局域网中，查看esp8266获取到的路由器地址

7. 将esp8266与电脑连接断开，连接到移动电源上

8. 在浏览器中打开esp8266获取到的局域网地址，通过拖动滑块来控制机械臂

##### 代码详解

\* WiFi机械臂程序

```arduino {.line-numbers}

#include <esp8266WiFi.h>

#include <FS.h>

#include <Servo.h>

#include "server.h"

//被连接WiFi的名称和密码

const char\* WIFI\_SSID = "AI";

const char\* WIFI\_PASSWORD = "raspberry";

Servo servos[9];

uint8\_t servo\_pins[9] = {D0,D1,D2,D3,D4,D5,D6,D7,D8};

uint8\_t count = 9;

void setAngle(uint8\_t di,uint8\_t vi){

if(di< 9 && vi < 180)

servos[di].write(vi);

}

void attachServos(){

for (uint8\_t i = 0; i < count; ++i){

servos[i].attach(servo\_pins[i]);

}

}

void detachServos(){

for (uint8\_t i = 0; i < count; ++i){

servos[i].detach();

}

}

void handleNotFound(){

String message = "File Not Found\n\n";

message += "URI: ";

message += server.uri();

message += "\nMethod: ";

message += (server.method() == https\_GET)?"GET":"POST";

message += "\nArguments: ";

message += server.args();

message += "\n";

for (uint8\_t i=0; i<server.args(); i++){

message += " " + server.argName(i) + ": " + server.arg(i) + "\n";

}

server.send(404, "text/plain", message);

}

void setup() {

// init the serial

Serial.begin(115200);

Serial.println();

Serial.println("Server initial");

//init the server

SPIFFS.begin();

Serial.println("SPIFFS begin");

attachServos();

Serial.println("Servos attached");

server.serveStatic("/", SPIFFS, "/index.html");

server.on("/set-servo", [](){

String uri = server.uri();

Serial.println(uri);

String di = server.arg("di");

String vi = server.arg("vi");

Serial.println("di="+di+" vi="+vi);

setAngle(di.toInt(),vi.toInt());

server.send(200, "text/plain", String("set to ")+"di="+di+" vi="+vi);

});

server.onNotFound(handleNotFound);

server.begin();

Serial.println("http server started");

// init the WiFi connection

Serial.println();

Serial.println();

Serial.print("INFO: Connecting to ");

WiFi.mode(WIFI\_STA);

Serial.println(WIFI\_SSID);

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("INFO: WiFi connected");

Serial.print("INFO: IP address: ");

Serial.println(WiFi.localIP());

}

void loop() {

server.handleClient();

}

```

### Part 1.4 esp32网络摄像头与人脸识别

esp32是esp8266的升级版本。拥有更强的处理能力，能够很好的处理实时视频和音频等数据。通过本部分来为小车增加实时视频的功能。

也许，你会希望在小车上装一个摄像头，这样就可以身临其境的遥控它了。

#### \*\*学习流程\*\*

##### 课程引入 （15分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[活动目标](#活动目标-3) | 实现实时网络摄像头的人脸识别功能 | 5分钟

[背景知识](#背景知识-3) | 了解esp32的相关知识与操作 | 10分钟

</center>

##### 基本任务 （35分钟）

<center>

活动名称 | 活动内容 | 时间分配

:-: | :-: | :-:

[硬件准备](#硬件准备-3) | 将硬件按文档进行连接 | 5分钟

[程序及操作](#程序及操作-3) | 完成程序及操作文档部分 | 30分钟

</center>

##### 活动总结 （10分钟）

#### \*\*活动目标\*\*

\* 确定活动方向并提出需要解决的问题

\* 了解esp32的功能及引脚并熟练了解实验步骤

\* 按照操作步骤实际操作并完成小车实时摄像等相关功能

\* 对实验进行总结，并分析遇到的问题

#### \*\*背景知识\*\*

##### A

文字描述

##### B

ESP32是一系列低成本，低功耗的片上 微控制器系统，集成了Wi-Fi和双模蓝牙。ESP32包括双核和单核变体，包括内置天线开关，功率放大器，低噪声接收放大器，滤波器和电源管理模块。

#### \*\*硬件准备\*\*

##### 硬件清单

\* esp32主板

\* ov2640摄像头

\* USB转TTL编程器

\* 杜邦线

##### 硬件连接

<center><img src=https://md.hass.live/niji/2019-05-09-ESP32-CAM-wiring-FTDI1.webp></center>

###### 注意：`IO0`口需要和它边上的`GND`口用一根杜邦线连接到一起，这样才可以正常上传代码

#### \*\*程序及操作\*\*

##### 操作步骤

1.打开`learn-ai`文件夹，打开路径`chapter1/part1/esp32\_projects/esp32\_webcam`

2.将上图连接好后，将USB转TTL编程器插入电脑

3.使用Arduino IDE打开文件`esp32——webcam.ino`

4.配置esp32的上传环境如下图所示：

<center><img src="https://md.hass.live/niji/2019-05-09-Xnip2019-05-09\_18-34-06.png"></center>

5.上传完毕后，将`IO0`口需要和它边上的`GND`口杜邦线拔掉，按一下esp32主板上面的`reset`键

6.打开[路由器管理地址](http://192.168.0.1)，esp32此时应该已经加入到了局域网中，查看esp32获取到的路由器地址

7.在浏览器中打开esp32获取到的局域网地址，在左侧最下方选择`Start Stream`

<center><img src=https://md.hass.live/niji/2019-05-09-Xnip2019-05-09\_18-31-57.png></center>

##### 代码详解

\* esp32\_webcam.ino

```arduino {.line-numbers}

#include "esp\_camera.h"

#include <WiFi.h>

// Select camera model

// #define CAMERA\_MODEL\_WROVER\_KIT

//#define CAMERA\_MODEL\_ESP\_EYE

//#define CAMERA\_MODEL\_M5STACK\_PSRAM

//#define CAMERA\_MODEL\_M5STACK\_WIDE

#define CAMERA\_MODEL\_AI\_THINKER

#include "camera\_pins.h"

const char\* ssid = "AI";

const char\* password = "raspberry";

void startCameraServer();

void setup() {

Serial.begin(115200);

Serial.setDebugOutput(true);

Serial.println();

camera\_config\_t config;

config.ledc\_channel = LEDC\_CHANNEL\_0;

config.ledc\_timer = LEDC\_TIMER\_0;

config.pin\_d0 = Y2\_GPIO\_NUM;

config.pin\_d1 = Y3\_GPIO\_NUM;

config.pin\_d2 = Y4\_GPIO\_NUM;

config.pin\_d3 = Y5\_GPIO\_NUM;

config.pin\_d4 = Y6\_GPIO\_NUM;

config.pin\_d5 = Y7\_GPIO\_NUM;

config.pin\_d6 = Y8\_GPIO\_NUM;

config.pin\_d7 = Y9\_GPIO\_NUM;

config.pin\_xclk = XCLK\_GPIO\_NUM;

config.pin\_pclk = PCLK\_GPIO\_NUM;

config.pin\_vsync = VSYNC\_GPIO\_NUM;

config.pin\_href = HREF\_GPIO\_NUM;

config.pin\_sscb\_sda = SIOD\_GPIO\_NUM;

config.pin\_sscb\_scl = SIOC\_GPIO\_NUM;

config.pin\_pwdn = PWDN\_GPIO\_NUM;

config.pin\_reset = RESET\_GPIO\_NUM;

config.xclk\_freq\_hz = 20000000;

config.pixel\_format = PIXFORMAT\_JPEG;

//init with high specs to pre-allocate larger buffers

if(psramFound()){

config.frame\_size = FRAMESIZE\_UXGA;

config.jpeg\_quality = 10;

config.fb\_count = 2;

} else {

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12;

config.fb\_count = 1;

}

#if defined(CAMERA\_MODEL\_ESP\_EYE)

pinMode(13, INPUT\_PULLUP);

pinMode(14, INPUT\_PULLUP);

#endif

// camera init

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK) {

Serial.printf("Camera init failed with error 0x%x", err);

return;

}

sensor\_t \* s = esp\_camera\_sensor\_get();

//initial sensors are flipped vertically and colors are a bit saturated

if (s->id.PID == OV3660\_PID) {

s->set\_vflip(s, 1);//flip it back

s->set\_brightness(s, 1);//up the blightness just a bit

s->set\_saturation(s, -2);//lower the saturation

}

//drop down frame size for higher initial frame rate

s->set\_framesize(s, FRAMESIZE\_QVGA);

#if defined(CAMERA\_MODEL\_M5STACK\_WIDE)

s->set\_vflip(s, 1);

s->set\_hmirror(s, 1);

#endif

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

startCameraServer();

Serial.print("Camera Ready! Use 'http://");

Serial.print(WiFi.localIP());

Serial.println("' to connect");

}

void loop() {

// put your main code here, to run repeatedly:

delay(10000);

}

```

### Part 1.5 综合与进阶

试着把前面的功能整合到一个小车上，并在一个界面上对它们进行读取和控制。

#### \*\*参考\*\*

\* 代码在`learn-ai/chapter1/part1/part1.5`

\* 代码将`Part1.2`和`Part1.3`结合到了一段程序中，对应网页文件可以输入`Part1.4`中esp32获取到的摄像头地址，实现小车画面实时显示

\* 硬件连接部分可将`Part1.2`,`Part1.3`和`Part1.4`结合在一起

\* `Part1.4`部分只需将esp32上的`3.3V`和`GND`用杜邦线连接到esp8266扩展板舵机连接处的对应接口,如下图：

<center><img src=https://md.hass.live/niji/2019-05-09-Xnip2019-05-09\_19-30-22.png></center>

<center><font color=red>红色</font>-正极V<br>

<font color=brown>棕色</font>-负极G<br>

</center>

#### \*\*demo\*\*

<center><iframe src="http://hass.live:9002" width="800" height="600" scrolling="yes" frameborder="0" mozallowfullscreen webkitallowfullscreen allowfullscreen></iframe></center>

#### \*\*抓取大赛\*\*

每小组为一个队伍，按照规则比赛抓取物品

##### 比赛规则

\* sector 1

\* sector 2

\* sector 3

### 尾声

通过前面的例子，你已经可以使用无线网络的方式来让esp8266读取传感器数据，控制简单的电机，查看网络摄像头等。其实这些已经是物联网的雏形。

在后面的章节里，结合强大的人工智能，你的小车会愈发强大。

通过语音技术，让小车听懂你的指令

通过计算机视觉，让机械臂自动识别和抓取特定物体

通过深度学习，让小车自动追踪特定物体，以及无人驾驶

---

## \*\*Part 2 物联网开源平台Home Assistant创意应用\*\*

Home Assistant是一个开源的物联网平台，兼容各种物联网协议。可以方便的接入和控制各种设备

这一部分主要在树莓派（Raspberry Pi）上面进行

树莓派上的Python环境使用 miniconda 进行管理

树莓派是运行Linux操作系统的微型卡片电脑，拥有GPIO端口与其他硬件通信，也具有HDMI，USB端口连接显示设备等

树莓派的操作系统在SD卡上，分发给同学的SD卡默认安装了所有的环境和依赖，相关的文档在桌面上

### Part 2.1 Home Assistant安装和配置

本课程采用[Docker](https://baike.baidu.com/item/Docker/13344470?fr=aladdin)方式进行安装

#### \*\*硬件准备\*\*

##### 硬件清单

\* 树莓派

##### 硬件连接

<center><img src=https://md.hass.live/404.gif></center>

#### \*\*程序及操作\*\*

##### 安装过程

在树莓派上打开终端，执行

```bash {.line-numbers}

//安装Docker

curl -ssl https://get.docker.com | sh

//安装指定版本的Home Assistant，本部分指定版本为0.82.0

sudo docker run --init -d --name="home-assistant" -v /home/pi/homeassistant:/config -v /etc/localtime:/etc/localtime:ro --net=host homeassistant/raspberrypi3-homeassistant:0.82.0

//启动Docker镜像

sudo docker start home-assistant

//进入镜像

sudo docker exec -it home-assistant env LANG=C.UTF-8 /bin/bash

```

正常情况下显示应和下图类似：

<center><img src="https://md.hass.live/Xnip2019-05-07\_18-35-34.png"></center>

本处即为Home Assistant的配置文件

主要配置在`configuration.yaml`中

文件夹`custom\_components`存放自定义组件

### Part 2.2 Home Assistant控制esp8266彩色灯

发放三国杀

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 2.3 Home Assistant人脸解锁

调用百度在线人工智能服务，进行人脸识别和语音播报，识别到注册人脸后自动把小灯打开。

#### \*\*硬件准备\*\*

\* 运行`Home Assistant`和`百度识别自定义组件`的树莓派

\* 安卓手机（`网络摄像头安卓APP`，提供快速稳定的视频服务。`kodi媒体中心安卓APP`，提供语音服务功能）

\* 手机也可替换为Part1.4所用的esp32摄像头

#### \*\*程序及操作\*\*

---

## \*\*Part 3 进阶项目-物联网机器人小绿\*\*

简介

### Part 3.1 组装一个小绿机器人

组装一个人形双足机器人，或仍然使用Part 1的小车

如果使用后者，跳过本部分至3.2

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 3.2 训练语音识别，开始和小绿聊天

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 3.3 让小绿听你指挥，控制一切

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 3.4 使用Google Blockly来控制小绿

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 3.5 使用OpenPose让小绿实时模仿你的动作

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

---

## \*\*Chapter 2 人工智能与机器人\*\*(jetson | 树莓派+PC)

这一部分包括……

## \*\*Part 1 人工智能算法相关案例体验\*\*

介绍

### Part 1.1 Tensorflow训练自定义图片分类器

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 1.2 使用RNN来生成古诗词

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

@import "./chapter2/part1/古诗词/poet\_rnn\_output.py" {cmd='/anaconda3/envs/learn-ai/bin/python'}

### Part 1.3 训练一个简单的游戏AI（Deep Q Network）

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 1.4 使用进化算法来训练超级马里奥

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 1.5 基于opencv在raspberry 上的人脸识别操作

在树莓派上利用opencv进行相关操作，实现对给定图片中相关人脸的识别功能

#### \*\*硬件准备\*\*

\* 树莓派、电源连接线、鼠标、键盘、以及显示屏

#### \*\*环境准备\*\*

1.首先我们需要安装python3

在这一步里我们需要登陆python官网

> <https://www.python.org/downloads/source/>

并下载合适的python版本

2.将树莓派开机并在命令行中输入如下操作

>sudo apt-get install libcv-dev

sudo apt-get install python-opencv

这一步我们是完成opencv在树莓派上的安装，在安装后我们可以通过以下操作来导入opencv

>1、打开命令行并输入

python3

![12](图片/1.png)

>2、接着输入

```bash {.line-numbers}

import cv2

```

![12](图片/2.png)

>3、之后我们可以通过以下代码来检测我们所使用的opencv版本

![12](图片/3.png)

3.接着我们需要安装两个库 (numpy 和 matlitlib) 一个用于计算一个用于图像绘制

使用以下代码在命令行中进行安装

```bash {.line-numbers}

pip install numpy

sudo apt-get install python3-matplotlib

```

这样我们就完成了实验所需所有的环境配置

#### \*\*程序及操作\*\*

1.首先需要在下面的网址里下载一个 cascade file

<https://raw.githubusercontent.com/shantnu/Webcam-Face-Detect/master/haarcascade\_frontalface\_default.xml>

并将其另存为

\*\_haarcascade\_frontalface\_default.xml\_\*

2.创建一个 .py 的文件 ( 可用我们之前安装的 Python3 打开 ) 并在文件中输入以下代码

```python {.line-numbers}

# Import OpenCV library

import cv2

# Load a cascade file for detecting faces

faceCascade = cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml");

# Load image

image = cv2.imread('picturex.jpeg')

# Convert into grayscale

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Look for faces in the image using the loaded cascade file

faces = faceCascade.detectMultiScale(gray, 1.2, 5)

for (x,y,w,h) in faces:

# Create rectangle around faces

cv2.rectangle(image,(x,y),(x+w,y+h),(255,255,0),2)

# Create the resizeable window

cv2.namedWindow('picturex', cv2.WINDOW\_NORMAL)

# Display the image

cv2.imshow('picturex', image)

# Wait until we get a key

k=cv2.waitKey(0)

# If pressed key is 's'

if k == ord('s'):

# Save the image

cv2.imwrite('convertedimage.jpg', image)

# Destroy all windows

cv2.destroyAllWindows()

# If pressed key is ESC

elif k == 27:

# Destroy all windows

cv2.destroyAllWindows()

```

\*\*\_第8/15/23行的picturex是我们需要更改的图片名字\_\*\*

3.将 cascade file、.py文件、以及我们所需要识别的图片发在同一路径下就像下图一样 ( 这保证了我们一会程序运行的确定性 )

<center>

![12](图片/4.png)

</center>

4.运行程序

需要我们在该目录下打开命令行（可使用快捷键鼠标点击路径按F4实现）

在命令行中输入

```python {.line-numbers}

#filename.py是你保存的代码文件的名称

python3 filename.py

```

之后我们就可以看到我们所选择的的图片加载出来，并且其中的人物的脸部会用方框圈起来啦

### Part 1.6 Neural Style 图像风格迁移

<center>

![Neural Style 图像风格迁移](http://psqevyxc3.bkt.clouddn.com/neuralstyle.jpg)

</center>

在神经网络之前，图像风格迁移的程序有一个共同的思路：分析某一种风格的图像，给那一种风格建立一个数学或者统计模型，再改变要做迁移的图像让它能更好的符合建立的模型。这样做出来效果还是不错的，但一个很大的缺点：\*\*一个程序基本只能做某一种风格或者某一个场景\*\*。因此基于传统风格迁移研究的实际应用非常有限。

而 Neural Style 程序通过输入一张代表内容的图片和一张代表风格的图片，使用深度学习网络输出一张融合了这个风格和内容的新作品。

#### \*\*准备\*\*

##### VGG 网络

[训练好的 VGG 19 网络](http://www.vlfeat.org/matconvnet/models/imagenet-vgg-verydeep-19.mat)，下载到项目文件夹“Neural Style 图像风格迁移”的中，或在运行时使用参数 `--network` 指定其位置。

##### 环境准备

除了在本课程最开始已经在 Anaconda 的 learn-ai 环境中安装好的 `TensorFlow` 外，还需安装 `Pillow` 软件包。

1. macOS 用户打开终端，Windows用户打开 Anaconda Prompt；

2. 输入下面的命令来进入 learn-ai 环境；

```bash {.line-numbers}

conda activate learn-ai

```

3. 安装 `Pillow` 软件包；

```bash {.line-numbers}

conda install pillow

```

4. 切换工作路径到项目文件夹；

```bash {.line-numbers}

cd 项目文件夹的路径

```

#### \*\*程序及操作\*\*

```bash {.line-numbers}

python neural\_style.py --content <输入图片> --styles <风格图片> --output <输出文件名> -- --iterations <迭代次数>

```

我们使用 examples 文件夹中的 1-content.jpg 和 2-style1.jpg 来举例，把上面命令中：

<输入图片> 替换为 `examples/1-content.jpg`

<风格图片> 替换为 `examples/2-style1.jpg`

<输出文件名> 替换为 `examples/output.jpg`，当然也可以不叫output，使用你自己喜欢的名字；

<迭代次数> 替换为 `100`

替换后的命令为：

```bash {.line-numbers}

python neural\_style.py --content examples/1-content.jpg --styles examples/2-style1.jpg --output examples/output.jpg -- --iterations 100

```

\*\*注意：\*\*\*<输入图片>\* 不建议使用过大的图片，这会明显的增加机器的负担，特别是对于性能差或者没有独立显卡的机器，可能需要数个小时来生成新的图片。\*<迭代次数>\* 在课程中使用的是 100 次迭代，可以初步看出机器风格迁移的效果。一般来说 1000 次迭代可以获得不错的图像质量，但会花费更多的时间。

<center>

![完成]( http://psqevyxc3.bkt.clouddn.com/finishProcess.jpg)

</center>

当窗口中的迭代次数变为 “100/100” 时就完成了整个图片的处理过程，打开 examples 文件夹中的图片就可以开到结果。

## \*\*Part 2 自动避障小车\*\*

项目介绍

### Part 2.1 环境准备与硬件搭建

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 2.2 通过超声波传感器进行避障

使用OpenCV来识别圆形物体，若识别到则发送串口指令给Arduino。Arduino控制机械臂进行抓取

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

---

## \*\*Part 3 自动追踪小车\*\*

使用OpenCV作为数据处理和输入，电机执行对应动作

### Part 3.1 环境准备与硬件搭建

简单的项目描述

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 3.2 OpenCV机械臂自动抓取特定形状物体

使用OpenCV来识别圆形物体，若识别到则发送串口指令给Arduino。Arduino控制机械臂进行抓取

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

### Part 3.3 OpenCV分类器训练，让小车追踪特定物体

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

#### \*\*硬件准备\*\*

#### \*\*程序及操作\*\*

---

## \*\*Part 4 无人驾驶小车\*\*

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

### Part 4.1 环境准备与硬件搭建

需要安装一些Python库并正确连接小车

#### \*\*硬件准备\*\*

##### 硬件清单

\* 树莓派

\* 树莓派电机扩展板

\* 摄像头

\* 小车套件

##### 硬件连接

<center><img src=https://md.hass.live/niji/2019-05-07-Xnip2019-05-07\_15-41-17.png?imageView2/0/interlace/1/q/46|imageslim></center>

#### \*\*程序及操作\*\*

##### 操作步骤

1.通过HDMI显示器连接树莓派，并使用USB键盘鼠标进行控制

2.在桌面上面找到名字为`learn-ai`的文件夹,打开路径`chapter2/part3/self\_driving\_car/`

3.在桌面按`ctrl`+`alt`+`T`来打开终端

4.打开终端后，执行以下命令

```bash

sudo apt update

install python3 ???

sudo apt install sklearn ???

```

### Part 4.2 电机和摄像头驱动测试

测试软硬件环境的安装是否正确

#### \*\*程序及操作\*\*

##### 操作步骤-电机测试

1.打开桌面上的`learn-ai`文件夹，打开路径`chapter2/part3/self\_driving\_car/`

2.打开终端，执行以下命令

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd computer

sudo python3 drive\_api2.py -s 150 //-s 150作为可选的参数，来指定行驶速度。可选范围是0-256

```

3.打开树莓派上的网络浏览器，在地址栏输入路由器管理地址，查看树莓派IP地址

4.在浏览器地址栏输入树莓派IP:81/drive,例如（192.168.0.100:81/drive）

5.在打开的界面上按键盘上的上下左右方向键来测试小车

6.测试完毕后，在终端输入`Ctrl`+`C`来结束当前任务

##### 操作步骤-摄像头测试

1.打开桌面上的`learn-ai`文件夹，打开路径`chapter2/part3/self\_driving\_car/`

2.打开终端，执行以下命令

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd test

sudo python3 stream\_server\_test.py

```

3.新建一个终端窗口

<center><img src="https://md.hass.live/terminal.png"></center>

4.在新的终端窗口中输入以下命令，如果有正常的视频画面输出，则测试通过

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd raspberryPi

sudo python3 stream\_client.py

```

5.在终端输入`Ctrl`+`C`来结束当前任务

### Part 4.3 无人驾驶数据采集及训练

搭建起车道，然后运行相应的收集数据的程序，按下笔记本的方向控制小车行驶，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

#### \*\*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

<center><img src=https://md.hass.live/track.jpg></center>

#### \*\*程序及操作\*\*

##### 操作步骤-驾驶数据采集

1.打开`learn-ai`文件夹，打开路径`chapter2/part3/`

2.打开终端，执行以下命令

```bash

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd computer

sudo python3 collect\_training\_data2.py

```

3.新建一个终端窗口，输入

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd raspberryPi

sudo python3 stream\_client.py

```

4.描述训练过程，按`q`结束训练

##### 操作步骤-驾驶数据训练

1.新建一个终端窗口，输入

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd computer

sudo python3 model\_training.py

```

<center><img src='https://md.hass.live/404.gif'></center>

2.训练完成后，训练文件在`文件路径`

### Part 4.4 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

#### \*\*程序及操作\*\*

##### 操作步骤

1.打开终端，输入

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd computer

python3 rc\_drive\_nn\_only.py

```

2.新建一个终端窗口，输入

```bash {.line-numbers}

cd ~/Desktop/learn-ai/chapter2/part3/self\_driving\_car

cd raspberryPi

python stream\_client.py

```