

人工智能课程文档-课程活动部分

目录

- 环境准备
 - Anaconda
 - 基本命令概览
 - setup
 - VSCode
 - setup
 - CP2102驱动
 - Arduino IDE
 - setup
 - 其他
 - 下载课程所需文件
- Chapter 1 物联网与机器人（树莓派，esp8266&esp32）
 - Part 1 使用esp8266开发板读取和控制传感器、舵机和电机
 - Part 1.1 使用esp8266在网页上读取传感器数据，绘制实时变化曲线
 - 活动目标
 - 时间分配
 - 背景知识
 - 硬件准备
 - 程序及操作
 - Part 1.2 WiFi遥控小车
 - demo
 - 活动目标
 - 时间分配
 - 背景知识
 - 硬件准备
 - 程序及操作
 - Part 1.3 WiFi机械臂
 - demo
 - 活动目标
 - 时间分配
 - 背景知识
 - 硬件准备
 - 程序及操作
 - Part 1.4 esp32网络摄像头与人脸识别
 - 活动目标
 - 时间分配
 - 背景知识
 - 硬件准备
 - 程序及操作
 - Part 1.5 综合与进阶
 - 参考
 - demo
 - 抓取大赛
 - 尾声
 - 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
 - 硬件准备
 - 程序及操作
 - Part 2.2 Home Assistant控制esp8266彩色灯
 - 硬件准备
 - 程序及操作
 - Part 2.3 Home Assistant人脸解锁

- 硬件准备
- 程序及操作
- 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
 - 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
 - 硬件准备
 - 程序及操作
 - Part 3.3 让小绿听你指挥，控制一切
 - 硬件准备
 - 程序及操作
 - Part 3.4 使用Google Blockly来控制小绿
 - 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
 - 硬件准备
 - 程序及操作
- 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 1.1 Tensorflow训练自定义图片分类器
 - 硬件准备
 - 程序及操作
 - Part 1.2 使用RNN来生成古诗词
 - 硬件准备
 - 程序及操作
 - Part 1.3 训练一个简单的游戏AI（Deep Q Network）
 - 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
 - 硬件准备
 - 程序及操作
- Part 2 自动避障小车
 - Part 2.1 环境准备与硬件搭建
 - 硬件准备
 - 程序及操作
 - Part 2.2 通过超声波传感器进行避障
 - 硬件准备
 - 程序及操作
- Part 3 自动追踪小车
 - Part 3.1 环境准备与硬件搭建
 - 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
 - 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
 - 硬件准备
 - 程序及操作
- Part 4 无人驾驶小车
 - Part 4.1 环境准备与硬件搭建
 - 硬件准备
 - 程序及操作
 - Part 4.2 电机和摄像头驱动测试
 - 程序及操作
 - Part 4.3 无人驾驶数据采集及训练
 - 硬件准备
 - 程序及操作
 - Part 4.4 开始无人驾驶

- 程序及操作

环境准备

环境准备相关所有相关软件, 请[点击这里](#)来下载

所需的用户名和密码均为 `sli`

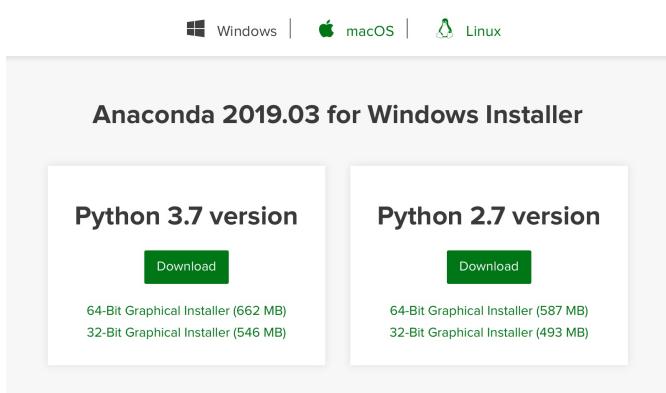
也可以选择在下方提供的官方网址下载

推荐按顺序依次安装以下软件, 以避免因依赖问题报错

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

```
1 | conda create -n 环境名字
2 | //例如:
3 | conda create -n py27 python=2.7
4 | //表示创建一个名字为py27, 运行python2.7的虚拟环境
5 | //后面的python=2.7是可选输入
6 | //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

```
1 | conda install 包名
2 | //例如: 安装OpenCV
3 | conda install opencv
```

其他命令

```
1 | //退出环境
2 | conda deactivate
3 | //列出环境
```

```
4 | conda-env list  
5 | //删除环境  
6 | 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)

```
1 | //创建一个名字为myenv的虚拟环境  
2 | conda create -n learn-ai  
3 | //激活myenv虚拟环境  
4 | conda activate learn-ai  
5 | //无独立显卡的电脑使用这条命令  
6 | conda install tensorflow  
7 | //有独立显卡的电脑使用这条命令  
8 | conda install tensorflow-gpu  
9 | //安装opencv  
10 | conda install opencv  
11 | //安装git命令  
12 | conda install git
```

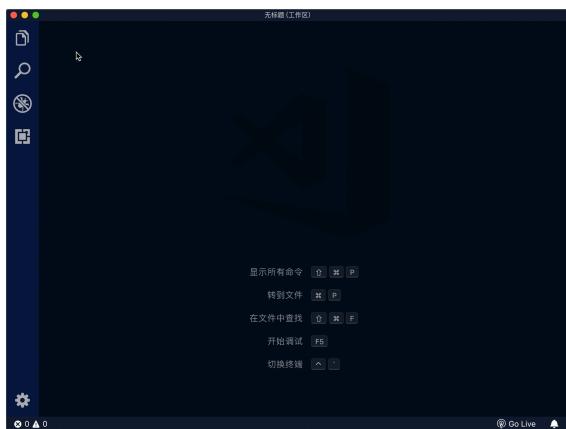
VSCode

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

https://www.silabs.com/documents/public/datasheets/CP2102-Datasheet.pdf

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>

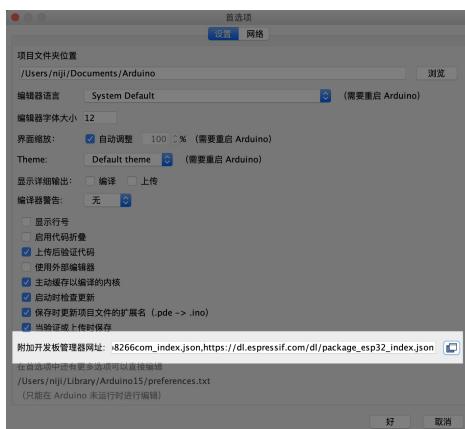
setup

1. 下载安装Arduino IDE

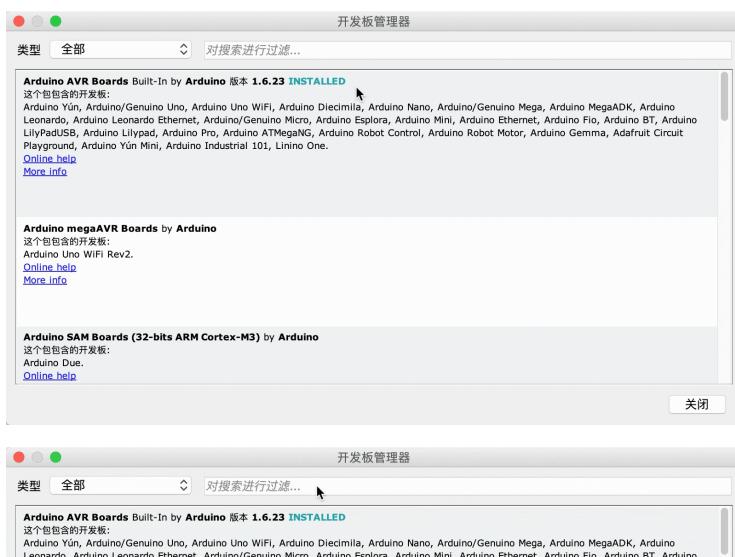
2. 在文件 - 首选项 - 附加开发板管理器网址 一栏中输

进入https://arduino.esp8266.com/stable/package_esp8266com_index.json

https://dl.espressif.com/dl/package_esp32_index.json 重启IDE

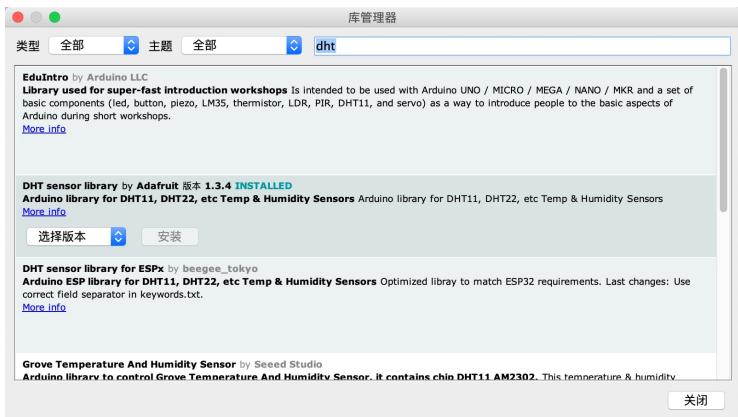


3. 在 **工具 - 开发板 - 开发板管理器** 中分别搜索esp8266和esp32,点击对应项进行安装





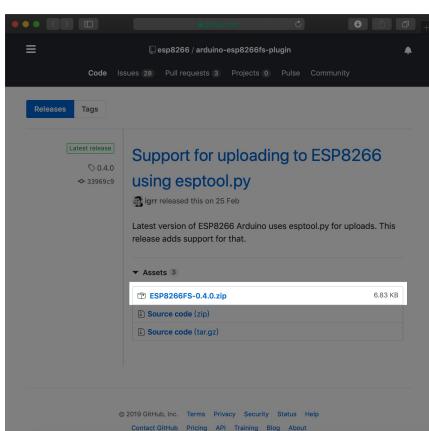
4. 在 工具 - 管理库 中搜索DHT,选择DHT sensor library by Adafruit



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:



7.设置开发板和端口



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

Xshell (Windows)、FinalShell (macOS)、Google Chrome、VNC

Viewer等

下载课程所需文件

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

```
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

Chapter 1 物联网与机器人 (树莓派, esp8266&esp32)

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

主要包括:

1. 使用物联网开发板来读取和控制传感器、灯和舵机等设备
2. 安装和配置物联网平台, 实现语音控制和人脸识别

Part 1 使用esp8266开发板读取和控制传感器、舵机和电机

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

Part 1.1 使用esp8266在网页上读取传感器数据，绘制实时变化曲线

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

活动目标

- 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

时间分配

- 活动准备：15分钟

活动名称	活动内容	时间分配
观看相关视频	了解物联网基本概念	5分钟
教师演示	对实验中涉及到的软硬件进行介绍	10分钟

- 活动过程：35分钟

活动名称	活动内容	时间分配
硬件准备	将硬件按文档进行连接	5分钟
程序及操作	教师引导完成程序及操作文档部分	30分钟

- 活动总结：10分钟

背景知识

物联网背景知识

开发板背景知识

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。

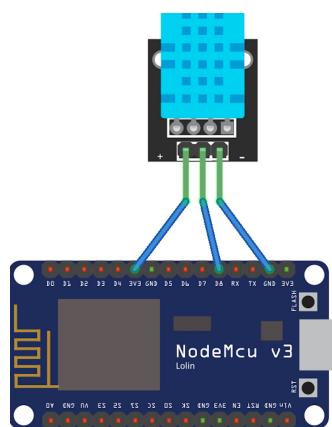
通过自带的GPIO口连接传感器，传感器将环境数据转化为电信号发送给esp8266读取、处理并输出。

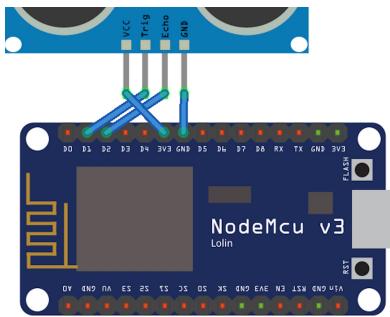
硬件准备

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





程序及操作

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](#) 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



```
sketch_may05c | Arduino 1.8.9
void setup() {
  // put your setup code here, to run once:
}

void loop() {
```

5. 打开 [路由器管理地址](#)，esp8266 此时应该已经加入了局域网中，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](#) 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

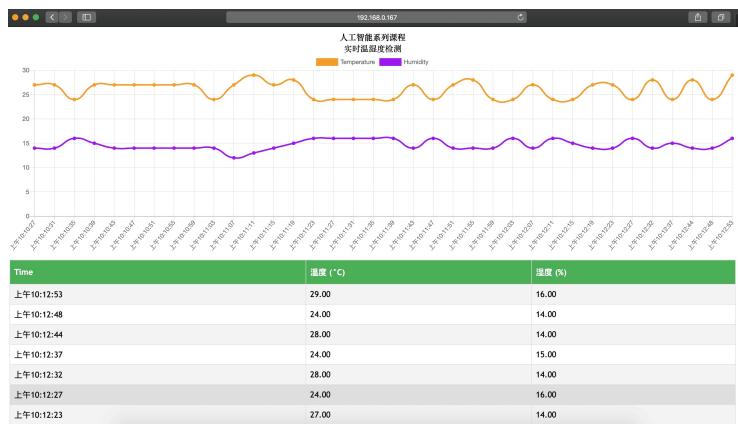


```
sketch_may05c | Arduino 1.8.9
void setup() {
  // put your setup code here, to run once:
}

void loop() {
```

5. 打开 [路由器管理地址](#)，esp8266 此时应该已经加入了局域网中，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



代码详解

- 温湿度传感器

```
55 //////////////////////////////////////////////////////////////////
56
57 // 初始化你连接的DHT传感器
58 DHT dht(DHTPin, DHTTYPE);
59 //定义两个float变量来读取温湿度
60 float Temperature; //温度
61 float Humidity; //湿度
62
63
64 //开始最后的准备工作
65 void setup() {
66   Serial.begin(115200);
67   delay(100);
68
69   pinMode(DHTPin, INPUT);
70   dht.begin();
71
72   Serial.println("Connecting to ");
73   Serial.println(ssid);
74
75   WiFi.begin(ssid, password);
76
77   //检查连接是否正常
78   while (WiFi.status() != WL_CONNECTED) {
79     delay(1000);
80     Serial.print(".");
81   }
82   Serial.println("");
83   Serial.println("WiFi connected..!");
84   Serial.print("Got IP: "); Serial.println(WiFi.local
85 IP());
86
87   server.on("/", handle_OnConnect);
88   server.onNotFound(handle_NotFound);
89
90   server.begin();
91   Serial.println("http server started");
92 }
93
94 void handle_OnConnect() {
95   Temperature = dht.readTemperature();
96   Humidity = dht.readHumidity();
97   server.send(200, "text/html", SendHTML(Temperature,H
98 umidity));
99 }
100
101 void handle_NotFound(){
102   server.send(404, "text/plain", "Not found");
103 }
104
105 String SendHTML(float Temperaturestat,float Humidityst
106 at){
107   String ptr = "<!DOCTYPE html> <html>\n";
108   ptr += "<head><meta charset=\"UTF-8\"><meta name=\"Vi
109 ewport\" content=\"width=device-width, initial-scale=1
110 .0, user-scalable=no\">\n";
111   ptr += "<title>esp8266 DHT11</title>\n";
112
113   ptr += "<script>\n";
114   ptr += "setInterval(loadDoc,200);\n";
115   ptr += "function loadDoc() {\n";
116   ptr += "var xhttp = new XMLHttpRequest();\n";
117   ptr += "xhttp.onreadystatechange = function() {\n";
118   ptr += "if (this.readyState == 4 && this.status == 20
119 0) {\n";
120   ptr += "document.getElementById(\"webpage\").innerHTML
121 =this.responseText}\n";
122   ptr += "};\n";
123   ptr += "xhttp.open(\"GET\", \"/\", true);\n";
124   ptr += "xhttp.send();\n";
125   ptr += "}\n";
126   ptr += "</script>\n";
127   ptr += "</head>\n";
128   ptr += "<body>\n";
```

```

129  ptr += "<div id=\"webpage\">\n";
130
131  ptr += "<p>温度: ";
132  ptr +=(int)Temperaturestat;
133  ptr += "°C</p>";
134  ptr += "<p>湿度: ";
135  ptr +=(int)Humiditystat;
136  ptr += "%</p>";
137
138  ptr += "</div>\n";
139  ptr += "</body>\n";
140  ptr += "</html>\n";
141  return ptr;
142 }

//所有准备工作就绪，开始工作
void loop() {
    server.handleClient();
}

```

- 超声波传感器

```

1 #include <Arduino.h>
2 #include <esp8266WiFi.h>
3 #include <esp8266WiFiMulti.h>
4 #include <esp8266httpsClient.h>
5 #include <esp8266WebServer.h>
6 // ultrasonic pinout
7 #define ULTRASONIC_TRIG_PIN      5    // pin TRIG to D1
8 #define ULTRASONIC_ECHO_PIN      4    // pin ECHO to D2
9
10 const char* WiFi_ssid = "AI";           // SSID
11 const char* WiFi_password = "raspberry"; // WI
12 FI
13 esp8266WebServer server(80);
14 esp8266WiFiMulti WiFiMulti;
15
16 void setup() {
17     Serial.begin(115200);
18     Serial.println("*****");
19     Serial.println("*****");
20     Serial.println("***** Program Start : Connect U
ltronics HC-SR04 + esp8266 to AskSensors over http");
21     Serial.println("Wait for WiFi... ");
22     Serial.print("***** connecting to WIFI : ");
23     Serial.println(WiFi_ssid);
24     WiFi.begin(WiFi_ssid, WiFi_password);
25     while (WiFi.status() != WL_CONNECTED) {
26         delay(500);
27         Serial.print(".");
28     }
29     Serial.println("");
30     Serial.println("-> WiFi connected");
31     Serial.println("-> IP address: ");
32     Serial.println(WiFi.localIP());
33     // ultraonic setup
34     pinMode(ULTRASONIC_TRIG_PIN, OUTPUT);
35     pinMode(ULTRASONIC_ECHO_PIN, INPUT);
36     server.on("/", handle_OnConnect);
37     server.onNotFound(handle_NotFound);
38     server.begin();
39 }
40
41 void handle_OnConnect() {
42     long duration, distance;
43     digitalWrite(ULTRASONIC_TRIG_PIN, LOW);
44     delayMicroseconds(2);
45     digitalWrite(ULTRASONIC_TRIG_PIN, HIGH);
46     delayMicroseconds(10);
47     digitalWrite(ULTRASONIC_TRIG_PIN, LOW);
48     duration = pulseIn(ULTRASONIC_ECHO_PIN, HIGH);
49     distance = (duration/2) / 29.1;
50     server.send(200, "text/html", SendHTML(distance));
51 }

```

```

52 }
53
54 void handle_NotFound(){
55   server.send(404, "text/plain", "Not found");
56 }
57
58 String SendHTML(long distance){
59   String ptr = "<!DOCTYPE html> <html>\n";
60   ptr += "<head><meta charset=\"UTF-8\"><meta name=\"vi
61 ewport\" content=\"width=device-width, initial-scale=1
62 .0, user-scalable=no\">\n";
63   ptr += "<title>esp8266 DHT11</title>\n";
64
65   ptr += "<script>\n";
66   ptr += "setInterval(loadDoc,200);\n";
67   ptr += "function loadDoc() {\n";
68   ptr += "var xhttp = new XMLHttpRequest();\n";
69   ptr += "xhttp.onreadystatechange = function() {\n";
70   ptr += "if (this.readyState == 4 && this.status == 20
71 0) {\n";
72   ptr += "document.getElementById(\"webpage\").innerHTML
73  =this.responseText}\n";
74   ptr += "};\n";
75   ptr += "xhttp.open(\"GET\", \"/\", true);\n";
76   ptr += "xhttp.send();\n";
77   ptr += "}\n";
78   ptr += "</script>\n";
79   ptr += "</head>\n";
80   ptr += "<body>\n";
81   ptr += "<div id=\"webpage\">\n";
82
83   ptr += "<p>距离: ";
84   ptr += (float)distance;
85   ptr += "厘米</p>";
86
87   ptr += "</div>\n";
88   ptr += "</body>\n";
89   ptr += "</html>\n";
90   return ptr;
91 }
92
93 void loop() {
94   server.handleClient();
95 }
```

- 温湿度传感器变化曲线

```

1 #include <ESP8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <ESP8266WebServer.h>
4
5 #include "index.h" //Our HTML webpage contents with ja
6 vascripts
7 #include "DHTesp.h" //DHT11 Library for ESP
8
9 #define LED 2          //On board LED
10 #define DHTpin 15      //D8 of NodeMCU is GPIO15
11
12 DHTesp dht;
13
14 const char* ssid = "AI";
15 const char* password = "raspberry";
16
17 ESP8266WebServer server(80); //Server on port 80
18
19 void handleRoot() {
20   String s = MAIN_page; //Read HTML contents
21   server.send(200, "text/html", s); //Send web page
22 }
23
24 float humidity, temperature;
25
```

```

26 void handleADC() {
27     int a = analogRead(A0);
28
29     String data = "{\"ADC\":"+String(a)+"\", \"Temperat";
30     ure\":\""+ String(temperature) +"\", \"Humidity\":\""+
31     String(humidity) +"\"}";
32
33     digitalWrite(LED,!digitalRead(LED)); //Toggle LED on
34     data request ajax
35     server.send(200, "text/plain", data); //Send ADC valu;
36     e, temperature and humidity JSON to client ajax request
37
38
39     //Get Humidity temperatue data after request is compl;
40     ete
41     //Give enough time to handle client to avoid problems
42     delay(dht.getMinimumSamplingPeriod());
43
44     humidity = dht.getHumidity();
45     temperature = dht.getTemperature();
46
47     Serial.print(humidity, 1);
48     Serial.print(temperature, 1);
49     Serial.print(dht.toFahrenheit(temperature), 1);
50 }
51
52 //=====
53 =====
54 //           SETUP
55 //=====
56 =====
57 void setup()
58 {
59     Serial.begin(115200);
60     Serial.println();
61
62     dht.setup(DHTpin, DHTesp::DHT11); //for DHT11 Connec;
63     t DHT sensor to GPIO 17
64     //dht.setup(DHTpin, DHTesp::DHT22); //for DHT22 Conn;
65     ect DHT sensor to GPIO 17
66
67     WiFi.begin(ssid, password);      //Connect to your Wi;
68     Fi router
69     Serial.println(")");
70
71     //Onboard LED port Direction output
72     pinMode(LED,OUTPUT);
73
74     // Wait for connection
75     while (WiFi.status() != WL_CONNECTED) {
76         delay(500);
77         Serial.print(".");
78     }
79
80     //If connection successful show IP address in serial
81     monitor
82     Serial.println(")");
83     Serial.print("Connected to ");
84     Serial.println(ssid);
85     Serial.print("IP address: ");
86     Serial.println(WiFi.localIP()); //IP address assign;
87     ed to your ESP
88
89     server.on("/", handleRoot);      //Which routine to
90     handle at root location. This is display page
91     server.on("/readADC", handleADC); //This page is cal;
92     led by java Script AJAX
93
94     server.begin();                //Start server
95     Serial.println("HTTP server started");
96 }
97
98 //=====
99 =====

```

```
//===== LOOP =====
void loop()
{
    server.handleClient(); //Handle client requests
}
```

Part 1.2 WiFi遥控小车

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

demo



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

时间分配

- 活动准备: 15分钟
- 活动过程: 30分钟
- 活动总结: 15分钟

背景知识

物联网背景知识

开发板背景知识

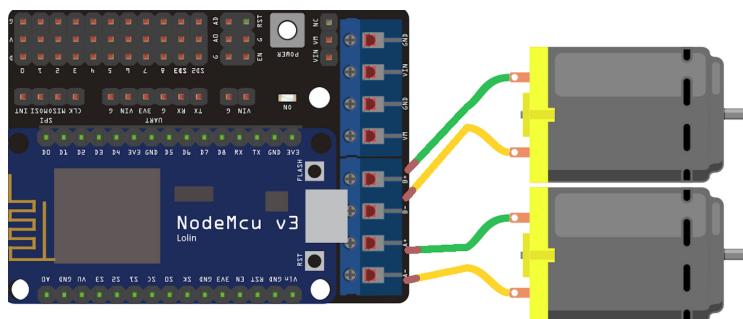
文字描述

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



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**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



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，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

```
1 #include <esp8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <esp8266WebServer.h>
4 #include <esp8266mDNS.h>
5 #include <FS.h>
6 #define Motor_AE D1      //Motor A/B,E enable,D Direct
7 ion
8 #define Motor_AD D3
9
10 #define Motor_BE D2
11 #define Motor_BD D4
12
13 #define R_AHEAD HIGH
14 #define L_AHEAD LOW
15
16 String command;
17
18 esp8266WebServer server(80);
19
20 const int led = 13;
21
22 void carInit(){
23     pinMode(Motor_AE, OUTPUT);
24     pinMode(Motor_AD, OUTPUT);
25     pinMode(Motor_BE, OUTPUT);
26     pinMode(Motor_BD, OUTPUT);
27
28     Serial.begin(115200);
29     Serial.println("Car begin");
30 }
31 void goAhead(){
32     digitalWrite(Motor_AE, HIGH);
33     digitalWrite(Motor_AD, L_AHEAD);
34     digitalWrite(Motor_BE, HIGH);
35     digitalWrite(Motor_BD, R_AHEAD);
36 }
37
38 void goBack(){
39     digitalWrite(Motor_AE, HIGH);
40     digitalWrite(Motor_AD, !L_AHEAD);
41     digitalWrite(Motor_BE, HIGH);
42     digitalWrite(Motor_BD, !R_AHEAD);
43 }
44
45 void goRight(){
46     digitalWrite(Motor_BE, LOW);
47     digitalWrite(Motor_AE, HIGH);
48     digitalWrite(Motor_AD, L_AHEAD);
49 }
50
51 void goLeft(){
52     digitalWrite(Motor_BE, HIGH);
53     digitalWrite(Motor_AE, LOW);
54     digitalWrite(Motor_BD, R_AHEAD);
55 }
56
57 void stopRobot(){
58     digitalWrite(Motor_AE, LOW);
59     digitalWrite(Motor_BE, LOW);
60 }
61
62 void handleNotFound(){
63     digitalWrite(led, 1);
```

```

63     digitalWrite(led, 1);
64     String message = "File Not Found\n\n";
65     message += "URI: ";
66     message += server.uri();
67     message += "\nMethod: ";
68     message += (server.method() == https_GET)?"GET":"POS
69     T";
70     message += "\nArguments: ";
71     message += server.args();
72     message += "\n";
73     for (uint8_t i=0; i<server.args(); i++){
74       message += " " + server.argName(i) + ":" + server.
75       arg(i) + "\n";
76     }
77     server.send(404, "text/plain", message);
78     digitalWrite(led, 0);
79   }
80
81 void setup(void){
82   carInit();
83   SPIFFS.begin();
84
85   uint8_t mac[WL_MAC_ADDR_LENGTH];
86   WiFi.softAPmacAddress(mac);
87   String macID = String(mac[WL_MAC_ADDR_LENGTH - 2], H
88   EX) + String(mac[WL_MAC_ADDR_LENGTH - 1], HEX);
89   macID.toUpperCase();
90
91   String AP_NameString = "Wifi Car - " + macID;
92
93   char AP_NameChar[AP_NameString.length() + 1];
94   memset(AP_NameChar, 0, AP_NameString.length() + 1);
95
96   for (int i = 0; i < AP_NameString.length(); i++)
97     AP_NameChar[i] = AP_NameString.charAt(i);
98
99 const char* ssid = "AI";
100 const char* password = "raspberry";
101 WiFi.mode(WIFI_STA);
102 WiFi.begin(ssid, password);
103 Serial.println("");
104
105 while (WiFi.status() != WL_CONNECTED) {
106   delay(500);
107   Serial.print(".");
108 }
109 Serial.println("");
110 Serial.print("Connected to ");
111 Serial.println(ssid);
112 Serial.print("IP address: ");
113 Serial.println(WiFi.localIP());
114 if (MDNS.begin("esp8266")) {
115   Serial.println("MDNS responder started");
116 }
117
118
119 //server.on("/", handleRoot);
120 server.serveStatic("/", SPIFFS, "/index.html");
121
122 server.on("/get", [](){
123   String uri = server.uri();
124   Serial.println(uri);
125   command = server.arg("command");
126   if(command == "forward")
127     goAhead();
128   else if(command == "backward")
129     goBack();
130   else if(command == "left")
131     goLeft();
132   else if(command == "right")
133     goRight();
134   else if(command == "stop")
135     stopRobot();
136   Serial.println(command);
137   // setColors(red.toInt(), green.toInt(), blue.toInt());

```

```
137 //      setCount(red.count(),green.count(),blue.count());
138 );
139     server.send(200, "text/plain", String("set to ")+c
140 ommand);
141 );
142     server.onNotFound(handleNotFound);
143     server.begin();
144     Serial.println("http server started");
145 }
```



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

控制你的机械臂

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

时间分配

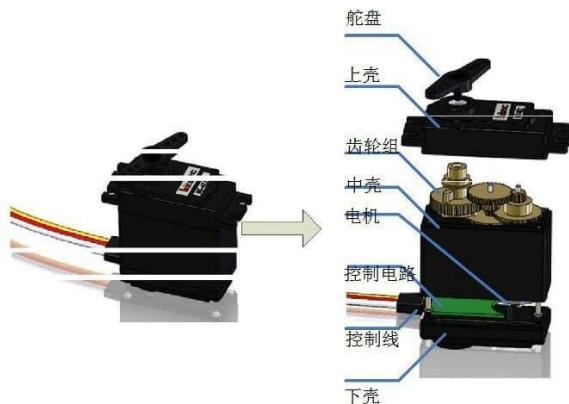
- 活动准备：10分钟
- 活动过程：35分钟
- 活动总结：15分钟

背景知识

舵机

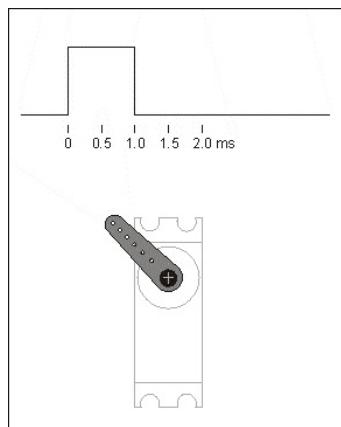
伺服电机通常被称为舵机，它是一种带有输出轴的小装置。当我们向伺服器发送一个控制信号时，输出轴就可以转到特定的位置。只要控制信号持续不变，伺服机构就会保持轴的角度位置不改变。如果控制信号发生变化，输出轴的位置也会相应发生变化。日常生活中，舵机常被用于遥控飞机、遥控汽车、机器人等。

舵机的工作原理



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。

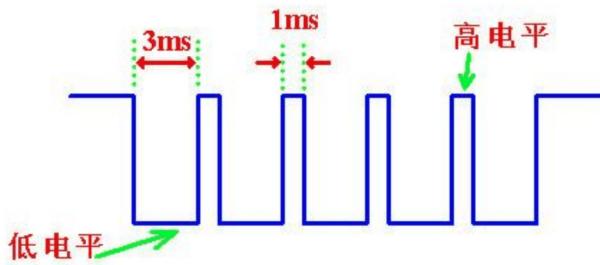


脉冲宽度调制（PWM）

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。



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：

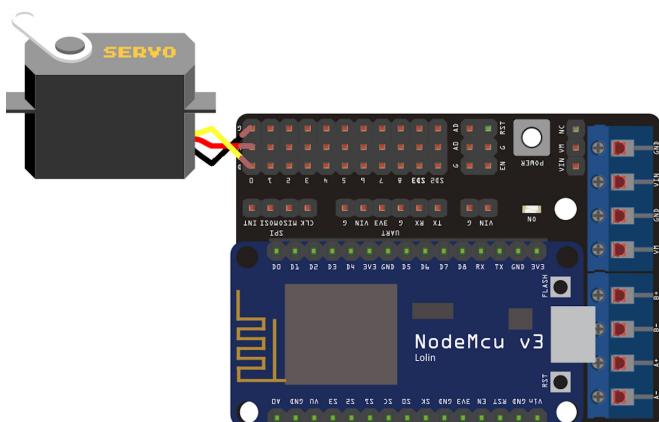
$$\frac{\text{高电平时间}}{\text{一个周期}} = \frac{1ms}{4ms} = 25\%$$

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



此处表示舵机连接到了D0口，最多可以连9个舵机 (D0-D9)

黄色-信号D (DATA)

红色-正极V (VCC)

棕色-负极G (GND)

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_https.ino
4. 记得把前面的环境准备部分再次确认，将环境正确配置，然后点击上

传按钮进行上传



```
sketch_may05c | Arduino 1.8.9
void setup() {
  // put your setup code here, to run once:
}

void loop() {
```

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

```
1 #include <esp8266WiFi.h>
2 #include <FS.h>
3 #include <Servo.h>
4 #include "server.h"
5
6 const char* WIFI_SSID = "AI";
7 const char* WIFI_PASSWORD = "raspberry";
8
9 Servo servos[9];
10 uint8_t servo_pins[9] = {D0,D1,D2,D3,D4,D5,D6,D7,D8};
11 uint8_t count = 9;
12 void setAngle(uint8_t di,uint8_t vi){
13   if(di< 9 && vi < 180)
14     servos[di].write(vi);
15 }
16 void attachServos(){
17   for (uint8_t i = 0; i < count; ++i){
18     servos[i].attach(servo_pins[i]);
19   }
20 }
21
22 void detachServos(){
23   for (uint8_t i = 0; i < count; ++i){
24     servos[i].detach();
25   }
26 }
27
28 void handleNotFound(){
29   String message = "File Not Found\n\n";
30   message += "URI: ";
31   message += server.uri();
32   message += "\nMethod: ";
33   message += (server.method() == https_GET)?"GET":"POS
34 T";
35   message += "\nArguments: ";
36   message += server.args();
37   message += "\n";
38   for (uint8_t i=0; i<server.args(); i++){
39     message += " " + server.argName(i) + ":" + server.
40     arg(i) + "\n";
41   }
42   server.send(404, "text/plain", message);
43 }
```

```

45 void setup() {
46     // init the serial
47     Serial.begin(115200);
48     Serial.println();
49     Serial.println("Server initial");
50     //init the server
51     SPIFFS.begin();
52     Serial.println("SPIFFS begin");
53     attachServos();
54     Serial.println("Servos attached");
55     server.serveStatic("/", SPIFFS, "/index.html");
56     server.on("/set-servo", [](){
57         String uri = server.uri();
58         Serial.println(uri);
59         String di = server.arg("di");
60         String vi = server.arg("vi");
61         Serial.println("di="+di+" vi="+vi);
62         setAngle(di.toInt(), vi.toInt());
63         server.send(200, "text/plain", String("set to ") +
64             di+" "+vi);
65     });
66     server.onNotFound(handleNotFound);
67     server.begin();
68     Serial.println("http server started");
69
70
71     // init the WiFi connection
72     Serial.println();
73     Serial.println();
74     Serial.print("INFO: Connecting to ");
75     WiFi.mode(WIFI_STA);
76     Serial.println(WIFI_SSID);
77     WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
78
79     while (WiFi.status() != WL_CONNECTED) {
80         delay(500);
81         Serial.print(".");
82     }
83
84     Serial.println("");
85     Serial.println("INFO: WiFi connected");
86     Serial.print("INFO: IP address: ");
87     Serial.println(WiFi.localIP());
88
89 }
90
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。

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

时间分配

- 活动准备：15分钟

活动名称	活动内容	时间分配

观看实验相关视频	了解esp32及其相关功能	10分钟
教师演示	对实验中可能会遇到困难的操作进行实际演示	10分钟

- 活动过程：30分钟
- 活动总结：15分钟

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



注意：I00 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：



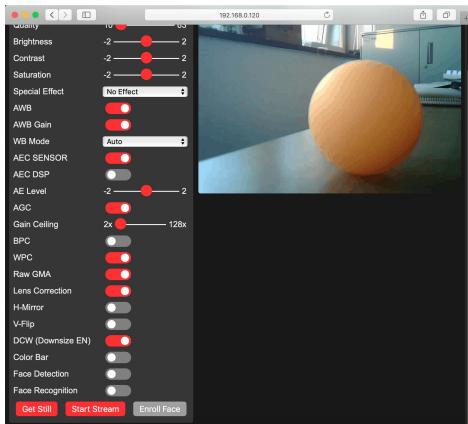
5. 上传完毕后，将 I00 口需要和它边上的 GND 口杜邦线拔掉，按一下

esp32主板上面的 reset 键

6. 打开[路由器管理地址](#)，esp32此时应该已经加入了局域网中，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



代码详解

- esp32_webcam.ino

```
1 #include "esp_camera.h"
2 #include <WiFi.h>
3
4
5
6 // Select camera model
7 // #define CAMERA_MODEL_WROVER_KIT
8 //##define CAMERA_MODEL_ESP_EYE
9 //##define CAMERA_MODEL_M5STACK_PSRAM
10 //##define CAMERA_MODEL_M5STACK_WIDE
11 #define CAMERA_MODEL_AI_THINKER
12
13 #include "camera_pins.h"
14
15 const char* ssid = "AI";
16 const char* password = "raspberry";
17
18 void startCameraServer();
19
20 void setup() {
21     Serial.begin(115200);
22     Serial.setDebugOutput(true);
23     Serial.println();
24
25     camera_config_t config;
26     config.ledc_channel = LEDC_CHANNEL_0;
27     config.ledc_timer = LEDC_TIMER_0;
28     config.pin_d0 = Y2_GPIO_NUM;
29     config.pin_d1 = Y3_GPIO_NUM;
30     config.pin_d2 = Y4_GPIO_NUM;
31     config.pin_d3 = Y5_GPIO_NUM;
32     config.pin_d4 = Y6_GPIO_NUM;
33     config.pin_d5 = Y7_GPIO_NUM;
34     config.pin_d6 = Y8_GPIO_NUM;
35
36     config.pin_d7 = Y9_GPIO_NUM;
37     config.pin_xclk = XCLK_GPIO_NUM;
38     config.pin_pclk = PCLK_GPIO_NUM;
39     config.pin_vsync = VSYNC_GPIO_NUM;
40     config.pin_href = HREF_GPIO_NUM;
41     config.pin_sscb_sda = SIOD_GPIO_NUM;
42     config.pin_sscb_scl = SIOC_GPIO_NUM;
43     config.pin_pwdn = PWDN_GPIO_NUM;
44     config.pin_reset = RESET_GPIO_NUM;
45     config.xclk_freq_hz = 20000000;
46     config.pixel_format = PIXFORMAT_JPEG;
47     //init with high specs to pre-allocate larger buffers
48
49     if(psramFound()){
50         config.frame_size = FRAMESIZE_UXGA;
```

```

50     config.jpeg_quality = 10;
51     config.fb_count = 2;
52 } else {
53     config.frame_size = FRAMESIZE_SVGA;
54     config.jpeg_quality = 12;
55     config.fb_count = 1;
56 }
57
58 #if defined(CAMERA_MODEL_ESP_EYE)
59     pinMode(13, INPUT_PULLUP);
60     pinMode(14, INPUT_PULLUP);
61#endif
62
63 // camera init
64 esp_err_t err = esp_camera_init(&config);
65 if (err != ESP_OK) {
66     Serial.printf("Camera init failed with error 0x%x",
67 err);
68     return;
69 }
70
71 sensor_t * s = esp_camera_sensor_get();
72 //initial sensors are flipped vertically and colors
73 are a bit saturated
74 if (s->id.PID == OV3660_PID) {
75     s->set_vflip(s, 1); //flip it back
76     s->set_brightness(s, 1); //up the brightness just a
77 bit
78     s->set_saturation(s, -2); //lower the saturation
79 }
80 //drop down frame size for higher initial frame rate
81 s->set_framesize(s, FRAMESIZE_QVGA);
82
83 #if defined(CAMERA_MODEL_M5STACK_WIDE)
84     s->set_vflip(s, 1);
85     s->set_hmirror(s, 1);
86#endif
87
88 WiFi.begin(ssid, password);
89
90 while (WiFi.status() != WL_CONNECTED) {
91     delay(500);
92     Serial.print(".");
93 }
94 Serial.println("");
95 Serial.println("WiFi connected");
96
97 startCameraServer();
98
99 Serial.print("Camera Ready! Use 'http://");
100 Serial.print(WiFi.localIP());
101 Serial.println("' to connect");
102 }
103
void loop() {
    // put your main code here, to run repeatedly:
    delay(10000);
}

```

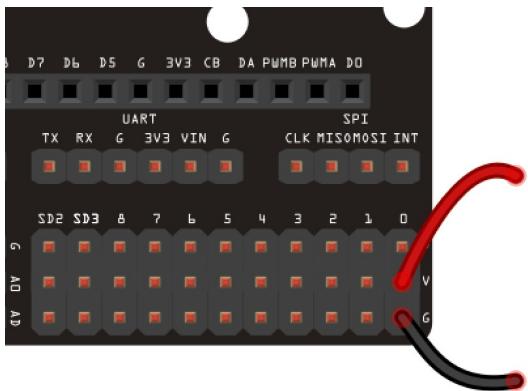
Part 1.5 综合与进阶

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:



红色-正极V

棕色-负极G

demo

Forward

Left

Stop

Right

Backward

在这里输入esp32的IP地址，以192开头

确定

D0

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方式进行安装

硬件准备

硬件清单

- 树莓派

硬件连接



程序及操作

安装过程

在树莓派上打开终端，执行

```
1 //安装Docker
2 curl -ssl https://get.docker.com | sh
3
4 //安装指定版本的Home Assistant, 本部分指定版本为0.82.0
5 sudo docker run --init -d --name="home-assistant" -v /
6 home/pi/homeassistant:/config -v /etc/localtime:/etc/
7 localtime:ro --net=host homeassistant/raspberrypi3-home
8 assistant:0.82.0
9
10 //启动Docker镜像
11 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：

```
pi@raspberrypi:~ $ sudo docker start home-assistant
home-assistant
pi@raspberrypi:~ $ sudo docker exec -it home-assistant /bin/bash
bash-4.4# ls
configuration.yaml      groups.yaml          home-assistant_v2.db-wal  tts
custom_components        home-assistant.log    images
customize.yaml          home-assistant_v2.db  scripts.yaml
deps                   home-assistant_v2.db-shm secrets.yaml
```

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 collections
import numpy as np
import tensorflow as tf

#-----数据预处理-----
# 

poetry_file = './chapter2/part1/古诗词/poetry.txt'

# 诗集
poetrys = []
with open(poetry_file, "r", encoding='utf-8') as f:
    for line in f:
        try:
            title, content = line.strip().split('：')
            content = content.replace(' ', '')
            if '「' in content or '（' in content or '〔' in content or '《' in content or '[' in content:
                continue
            if len(content) < 5 or len(content) > 79:
                continue
            content = '[' + content + ']'
            poetrys.append(content)
        except Exception as e:
            pass

# 按诗的字数排序
poetrys = sorted(poetrys, key=lambda line: len(line))
print('唐诗总数: ', len(poetrys))

# 统计每个字出现次数
all_words = []
for poetry in poetrys:
    all_words += [word for word in poetry]
counter = collections.Counter(all_words)
count_pairs = sorted(counter.items(), key=lambda x: -x[1])
words, _ = zip(*count_pairs)

# 取前多少个常用字
words = words[:len(words)] + (' ',)
# 每个字映射为一个数字ID
word_num_map = dict(zip(words, range(len(words))))
# 把诗转换为向量形式, 参考TensorFlow练习1
to_num = lambda word: word_num_map.get(word, len(words))
poetrys_vector = [list(map(to_num, poetry)) for poetry in poetrys]
#[[314, 3199, 367, 1556, 26, 179, 680, 0, 3199, 41, 506, 4
0, 151, 4, 98, 1],
#[339, 3, 133, 31, 302, 653, 512, 0, 37, 148, 294, 25, 54,
833, 3, 1, 965, 1315, 377, 1700, 562, 21, 37, 0, 2, 1253,
21, 36, 264, 877, 809, 1]
#....]

batch_size = 1
n_chunk = len(poetrys_vector) // batch_size
x_batches = []
y_batches = []
for i in range(n_chunk):
    start_index = i * batch_size
    end_index = start_index + batch_size

    batches = poetrys_vector[start_index:end_index]
```

```

length = max(map(len,batches))
xdata = np.full((batch_size,length), word_num_map[' '],
np.int32)
for row in range(batch_size):
    xdata[row,:len(batches[row])] = batches[row]
ydata = np.copy(xdata)
ydata[:, :-1] = xdata[:, 1:]
"""
xdata          ydata
[6,2,4,6,9]      [2,4,6,9,9]
[1,4,2,8,5]      [4,2,8,5,5]
"""
x_batches.append(xdata)
y_batches.append(ydata)

#-----RNN-----
#-----#
input_data = tf.placeholder(tf.int32, [batch_size, None])
output_targets = tf.placeholder(tf.int32, [batch_size, None])
# 定义RNN
def neural_network(model='lstm', rnn_size=128, num_layers=2):
    if model == 'rnn':
        cell_fun = tf.nn.rnn_cell.BasicRNNCell
    elif model == 'gru':
        cell_fun = tf.nn.rnn_cell.GRUCell
    elif model == 'lstm':
        cell_fun = tf.nn.rnn_cell.BasicLSTMCell

    cell = cell_fun(rnn_size, state_is_tuple=True)
    cell = tf.nn.rnn_cell.MultiRNNCell([cell] * num_layers,
state_is_tuple=True)

    initial_state = cell.zero_state(batch_size, tf.float32)

    with tf.variable_scope('rnnlm'):
        softmax_w = tf.get_variable("softmax_w", [rnn_size,
len(words)+1])
        softmax_b = tf.get_variable("softmax_b", [len(words
)+1])
        with tf.device("/cpu:0"):
            embedding = tf.get_variable("embedding", [len(
words)+1, rnn_size])
            inputs = tf.nn.embedding_lookup(embedding, input_
data)

        outputs, last_state = tf.nn.dynamic_rnn(cell, inputs,
initial_state=initial_state, scope='rnnlm')
        output = tf.reshape(outputs, [-1, rnn_size])

        logits = tf.matmul(output, softmax_w) + softmax_b
        probs = tf.nn.softmax(logits)
        return logits, last_state, probs, cell, initial_state

#-----生成古诗-----
# 使用训练完成的模型

# def gen_poetry():
#     def to_word(weights):
#         t = np.cumsum(weights)
#         s = np.sum(weights)
#         sample = int(np.searchsorted(t, np.random.rand(1)*
s))
#         return words[sample]

#     _, last_state, probs, cell, initial_state = neural_net
work()

#     with tf.Session() as sess:

```

```
#     sess.run(tf.initialize_all_variables())
#
#     saver = tf.train.Saver(tf.all_variables())
#     saver.restore(sess, 'poetry.module-49')
#
#     state_ = sess.run(cell.zero_state(1, tf.float32))
#
#     x = np.array([list(map(word_num_map.get, '['))])
#     [probs_, state_] = sess.run([probs, last_state], f
# eed_dict={input_data: x, initial_state: state_})
#     word = to_word(probs_)
#     #word = words[np.argmax(probs_)]
#
#     poem = ''
#     while word != ']':
#         poem += word
#         x = np.zeros((1,1))
#         x[0,0] = word_num_map[word]
#         [probs_, state_] = sess.run([probs, last_state
# ], feed_dict={input_data: x, initial_state: state_})
#         word = to_word(probs_)
#         #word = words[np.argmax(probs_)]
#     return poem
# print(gen_poetry())
#
# 藏头诗
def gen_poetry_with_head_and_type(head, type):
    if type != 5 and type != 7:
        print('The second para has to be 5 or 7!')
        return
    def to_word(weights):
        t = np.cumsum(weights)
        s = np.sum(weights)
        sample = int(np.searchsorted(t, np.random.rand(1)*s
))
        return words[sample]
    _, last_state, probs, cell, initial_state = neural_net
    work()
    with tf.Session() as sess:
        sess.run(tf.initialize_all_variables())
        saver = tf.train.Saver()
        saver.restore(sess, './chapter2/part1/古诗词/poetry
.module-49')
        poem = ''
        i = 0
        for the_word in head:
            flag = True
            while flag:
                state_ = sess.run(cell.zero_state(1, tf
.float32))
                x = np.array([list(map(word_num_map.get
, '['))])
                [probs_, state_] = sess.run([probs, la
st_state], feed_dict={input_data: x, initial_state: state_})
                sentence = the_word
                x = np.zeros((1, 1))
                x[0, 0] = word_num_map[sentence]
                [probs_, state_] = sess.run([probs, la
st_state], feed_dict={input_data: x, initial_state: state_})
                word = to_word(probs_)
                sentence += word
                while word!='. ':
                    x = np.zeros((1, 1))
                    x[0, 0] = word_num_map[word]
                    [probs_, state_] = sess.run([probs,
last_state], feed_dict={input_data: x, initial_state: sta
```

```
te_})  
    word = to_word(probs_)  
  
    sentence += word  
  
    if len(sentence) == 2 + 2 * type:  
        sentence += '\n'  
        poem += sentence  
        flag = False  
  
return poem  
  
print(gen_poetry_with_head_and_type("深度学习", 7))
```



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 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 自动泊车小车

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

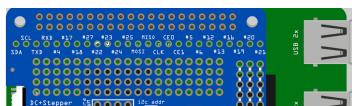
需要安装一些Python库并正确连接小车

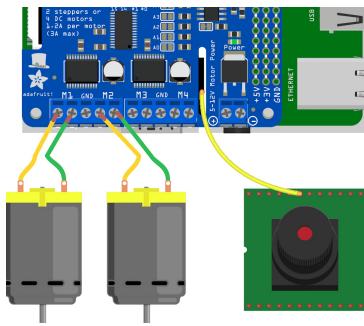
硬件准备

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





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

```
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

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd computer  
3 | 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
```

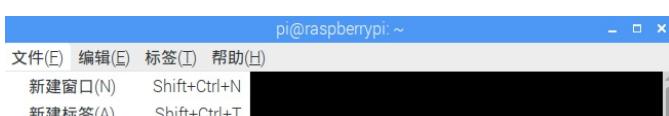
- 3.打开树莓派上的网络浏览器，在地址栏输入路由器管理地址，查看树莓派IP地址
- 4.在浏览器地址栏输入树莓派IP:81/drive,例如 (192.168.0.100:81/drive)
- 5.在打开的界面上按键盘上的上下左右方向键来测试小车
- 6.测试完毕后，在终端输入 `Ctrl + C` 来结束当前任务

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

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd test  
3 | sudo python3 stream_server_test.py
```

- 3.新建一个终端窗口





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

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd raspberryPi  
3 | 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



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

```
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

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd raspberryPi  
3 | 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

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd computer  
3 | sudo python3 model_training.py
```



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

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd computer  
3 | python3 rc_drive_nn_only.py
```

2.新建一个终端窗口，输入

```
1 | cd ~/Desktop/learn-ai/chapter2/part3/self_driving_car  
2 | cd raspberryPi  
3 | python stream_client.py
```