

# 人工智能自动驾驶 小车

Administrator

[公司名称][公司地址]

---

## 写在前面

自动驾驶智能小车项目基于 STEM 教育理念设计。本书作为项目攻略，旨在将项目拆分成拼图的碎片，项目书作为线索，希望读者能通过线索并参照攻略提取有效信息并加以组合分析完成此项目。

STEM 是科学(Science), 技术(Technology), 工程(Engineering), 数学(Mathematics) 四门学科英文首字母的缩写，其中科学在于认识世界、解释自然界的客观规律；技术和工程则是在尊重自然规律的基础上改造世界、实现对自然界的控制和利用、解决社会发展过程中遇到的难题；数学则作为技术与工程学科的基础工具。本攻略将从 S, T, E, M 四个方面展开，阅读时章节之间独立并不影响跳跃式阅读。面对如今信息爆炸的时代，编者希望读者能掌握如何从庞大的信息量中提取出有效的信息。本攻略虽然并无冗余的知识，但是对于完成单个项目任务书，如何从攻略中找出有效的信息就显得尤为关键。

总而言之，希望读者在完成项目的同时，体会从理论学习到实践的不同，鼓励读者能够通过进一步的学习优化项目，认识到找到解决问题的方法比解决问题更重要。

目录

写在前面 .....1

开发环境搭建 ..... 17

PYTHON 基础 ..... 33

    解释器 ..... 33

        调用 Python 解释器 ..... 33

        解释器的运行环境 ..... 34

    基本数据类型 ..... 35

    运算符 ..... 35

    数字 ..... 35

    字符串 ..... 35

    流程控制 ..... 35

运动控制项目书 ..... 37

初识传感项目书 ..... 40

光电循迹项目书 ..... 44

..... 错误!未定义书签。

算法循迹项目书 ..... 49

识别目标项目书 ..... 54

视觉循迹项目书 ..... 59

司机上路 ..... 64

## 一、 小车搭建

## 二、 开发环境搭建

## 三、 理论

### 1. 物理

#### 1) 矢量和标量

矢量是具有大小和方向的量，又称向量而标量只有大小没有方向。在给小车赋值时，speed 速度可以称为矢量，当 speed 的赋值为正时，电机正转，带动小车向前行进。赋值为负时，电机反转，带动小车向后运动。

#### 2) 参考系

#### 3) 车辆转向原理及基本转向方式

小车旋转的实质是两轮的旋转速度不同。相同时间内两轮的行驶路程不同，从而使得小车进行偏转。速度差越大，偏转的角度越大。相同速度时，原地转向效率最高。实际应用时，则需要根据需求的不同采取不同的转向方式



原地旋转	转圈	弧线前进
左电机正值，右电机负值。	左电机转动，右电机停止。	左电机高速转动，右边电机低速转动。
机器人原地旋转。	机器人以左轮为圆心转圈。	机器人沿弧线前进。

---

**原地转向：**两轮反向运动，即两个轮子的运动方向相反使小车旋转。

通过旁边的例图可以看出，反向旋转最接近原地旋转，这是因为此时小车的旋转中心在车轴上，当两个车轮的速度大小相等、方向相反时，小车的旋转中心刚好在车轴的正中心，实现中心在车轴上的原地旋转。相同情况下，这种旋转方式的旋转时间最短、速度相对更快，效率更高，可以保障车辆的安全，以最快的速度，最小的旋转半径通过弯道。

**单轮运动：**一个车轮的速度为零，另一个车轮的速度不为零，两者形成速度差，从而实现小车的偏转。这个时候的小车相当于一个旋转的圆规，速度为零的车轮代表固定的点，速度不为零的车轮代表运动的点，而我们给控制电机的速度大小代表相当于“圆规”画圆的快慢。此时小车做的是以固定轮为圆心的原地旋转。

**弧形前进：**两轮同向运动，但两轮的速度大小不同，形成速度差使小车旋转。如上图所示，这种速度一大一小的转弯，会使小车的转弯半径明显增大，也就是说当小车需要偏转较大的弧形时，用这种方式旋转更合适。

#### 4) 摩擦力

阻碍物体相对运动的力称为摩擦力，摩擦力的方向与物体相对运动的

---

方向相反。在编写程序使小车行进的过程中，会存在一定的摩擦因素导致赋给小车的值并不能使小车恰好完成预期目标，这时需要人为的调整参数，使小车正常完成任务。

#### 5) 超声波原理

### 2. 传感器

- 1) 触动传感器
- 2) 超声波传感器
- 3) 颜色传感器
- 4) 摄像头

### 3. 数学

- 1) 函数
- 2) 反函数
- 3) 积分
- 4) 微分
- 5) 平面直角坐标系
- 6) 空间直角坐标系

### 4. 图像处理背景知识

- 1) 小孔成像
- 2) 像素

- 
- 3) 分辨率
  - 4) 帧率
  - 5) 颜色空间

#### 四、 硬件知识

- 1) 小车搭建
- 2) 传感器安装

#### 五、 编程知识

##### 1. Python 基础

- 1) 解释器
- 2) 数据类型和变量

数据类型

整型(int)

介于-2147483648 和 21449483647 之间的整数都被认为是整型，更大的数被称作长整型，不过现在这两种类型已经合并，所以所有整数都可被认作整型。

浮点型(float)

数学中带有小数的实数代表浮点数。例如 3.2、-2.1。

如果要确定数值的分类，可以使用 Python 中内置的函数 `type`。

示例如下：

```
>>> type(1)
```

```
<class 'int'>
```

---

```
>>> type(1.0)
```

```
<class 'float'>
```

本质上，浮点数和整数的区别在于浮点数有小数部分。1.0、2.45、0.0056 这种含有小数部分的数都被当作浮点数。

## 虚数

在 Python 中还有一种专门面向工程师的数据类型，就是虚数。在数值后添加字母 `j`，并且数值和字母都不是字符串（不在引号中），python 就会将其自动理解为虚数类型。

示例如下：

```
>>> 365j
```

```
365j
```

## 字符串

字符串有多种形式，可以使用单引号（`'.....'`），双引号（`"....."`）都可以获得同样的结果。反斜杠 `\` 可以用来转义。

示例如下：

```
>>> " Hello ,world! "
```

```
' Hello ,world! '
```

```
>>> " Let's go! "
```

```
" Let's go! "
```

```
>>> ' Let\'s go! '
```

```
" Let's go! "
```

## 变量



---

在程序中总是显示地写字符串和数值是十分不便的，为了简化程序，可以给数据命名，之后通过名称引用数据。这种名称一般叫做变量。可以用一个等号将一个值与名称联系。

在 Python 中：

- 变量在第一次赋值时创建
- 变量在表达式中使用将被替代为它们的值
- 变量在表达式中使用以前必须已赋值
- 变量像对象一样不需要在一开始进行声明

也就是说，赋值会让变量自动生成。

```
>>>first=123
```

```
>>>second="this is a number"
```

```
>>>print(first)
```

```
123
```

```
>>>print(second)
```

```
"this is a number"
```

如果在 `print()` 函数里将一个变量用引号括起来，程序就会将他看作字符串而不是一个变量函数会打印这个变量的名称，而不是它的内容。

不是任何变量都可以用来命名，Python 把这类名称作为特殊的内置词，防止出现多义性。

And,as,assert,break,class,continue,def,del,else,except,exec,False,finally,for,from,global,if,import,in,is,lambda,not,None,or,padd,print,raise,ret

urn,try,True,while,with,yield

另外，变量名不可以树枝或多数非字母的字符开头（例如逗号、加减号和斜杠等），但下划线例外。

### 3) 运算符

运算符时由一个或多个字符组成的，用来表示某种晕眩地的符号，运算符与常量、变量、函数一起组成表达式，用来完成复杂的运算功能。

假设变量  $a=10$ ， $b=20$ 。

#### 算数运算符

运算符	描述	实例
+	加 - 两个对象相加	$a + b$ 输出结果 30
-	减 - 得到一个数减去另一个数	$a - b$ 输出结果 -10
*	乘 - 两个数相乘或是返回一个被重复若干次的字符串	$a * b$ 输出结果 200
/	除 - x 除以 y	$b / a$ 输出结果 2
%	取模 - 返回除法的余数	$b \% a$ 输出结果 0
**	幂 - 返回 x 的 y 次幂	$a ** b$ 为 10 的 20 次方， 输出结果

		1000000000000000000000
//	取整除 - 返回商的整数部分（向下取整）	>>> 9//2 4 >>> -9//2 -5

### 比较运算符

运算符	描述	实例
==	等于 - 比较对象是否相等	(a == b) 返回 False。
!=	不等于 - 比较两个对象是否不相等	(a != b) 返回 true。
<>	不等于 - 比较两个对象是否不相等	(a <> b) 返回 true。这个运算符类似 !=。
>	大于 - 返回 x 是否大于 y	(a > b) 返回 False。
<	小于 - 返回 x 是否小于 y。所有比较运算符返回 1 表示真，返回 0 表示假。这分别与特殊的变量 True 和 False 等价。	(a < b) 返回 true。
>=	大于等于 - 返回 x 是否大于等于 y。	(a >= b) 返回 False。
<=	小于等于 - 返回 x 是否小于等于 y。	(a <= b) 返回 true。

### 赋值运算符

运算符	描述	实例
=	简单的赋值运算符	c = a + b 将 a + b 的运算结果赋值为 c

+=	加法赋值运算符	c += a 等效于 c = c + a
-=	减法赋值运算符	c -= a 等效于 c = c - a
*=	乘法赋值运算符	c *= a 等效于 c = c * a
/=	除法赋值运算符	c /= a 等效于 c = c / a
%=	取模赋值运算符	c %= a 等效于 c = c % a
**=	幂赋值运算符	c **= a 等效于 c = c ** a

运算符	描述	实例
and	布尔"与" - 如果 x 为 False, x and y 返回 False, 否则它返回 y 的计算值。	(a and b) 返回 20。
or	布尔"或" - 如果 x 是非 0,它返回 x 的值, 否则它返回 y 的计算值。	(a or b) 返回 10。
not	布尔"非" - 如果 x 为 True, 返回 False 。如果 x 为 False, 它返回 True	not(a and b) 返回 False

	True。	
--	-------	--

逻辑运算符

Python 运算符优先级

运算符	描述
**	指数（最高优先级）
~ + -	按位翻转，一元加号和减号（最后两个的方法名为 +@ 和 -@）
* / % //	乘，除，取模和取整除
+ -	加法减法
>> <<	右移，左移运算符
&	位 'AND'
^	位运算符
<= < > >=	比较运算符
<> == !=	等于运算符
= %= /= //= -= += *= **=	赋值运算符
is is not	身份运算符
in not in	成员运算符
not and or	逻辑运算符

4). 条件控制

On 类及 Tank 类控制语句

类	函数声明	功能
Motor	<code>on(speed, brake=True, block=False)</code>	电机按指定速度一直旋转
LargeMo	<code>on_for_seconds(speed, seconds, b</code>	电机按指定速度旋转指

Motor	<i>on_for_degrees(speed, degrees, brake=True, block=True)</i>	电机按指定速度旋转指定角度
	<i>on_for_rotations(speed, rotations, brake=True, block=True)</i>	电机按指定速度旋转指定圈数
	<i>on_to_position(speed, position, brake=True, block=True)</i>	电机按指定速度旋转指定位置
	<i>on(left_speed, right_speed)</i>	电机组按指定速度一直旋转
MoveTank	<i>on_for_seconds(left_speed, right_speed, seconds, brake=True, block=True)</i>	电机组按指定速度旋转指定时间
	<i>on_for_degrees(left_speed, right_speed, degrees, brake=True, block=True)</i>	电机组按指定速度旋转指定角度
	<i>on_for_rotations(left_speed, right_speed, rotations, brake=True, block=True)</i>	电机组按指定速度旋转指定位置

## 5) 循环语句

### for 循环

for 语句在循环中常与内置函数 range 配合起来一起使用，表示取

---

一个范围内的整数。Range ( ) 中参数必须为整数，range (a,b) 表示初始值为 a,范围为  $a \leq i < b$ ，如果 range (b)，默认初始值为 0，表示范围为  $0 \leq i < b$ 。实现循环语句时，在循环结构里的语句等级比循环结构低一个等级。体现在结构上，循环的语句需要增加空格，习惯上增加四个空格（一个 Tab 键）。

for 循环的基本结构如下：

	变量	取值范围
	↑	↑
for i in range(数字):	for i in range (1,4)	
循环语句	循环语句	

例如 for i in range(4):

    循环语句

表示循环语句循环执行四次

Range(4)表示的范围是  $0 \leq i < 4$ ，配合 for 语句表示循环四次  
(0/1/2/3)

Range (1, 4) 表示的范围是  $1 \leq i < 4$ ,配合 for 语句表示循环三次  
(1/2/3)

While 循环

while 语句用于在表达式保持为真的情况下重复地执行：

---

while （表达式）：

### 循环语句

这将重复地检验表达式，并且如果其值为真就执行循环语句；如果表达式值为假（这可能在第一次检验时就发生）则如果 else 子句体存在就会被执行并终止循环。

循环语句中的 break 语句在执行时将终止循环且不执行 else 子句体。循环语句中的 continue 语句在执行时将跳过子句体中的剩余部分并返回检验表达式。

6) Break

7) Continue

8) 列表

9) 元组

10) 字典

11) 集合

12) 函数



---

上海师范大学—“自动驾驶”小车项目组

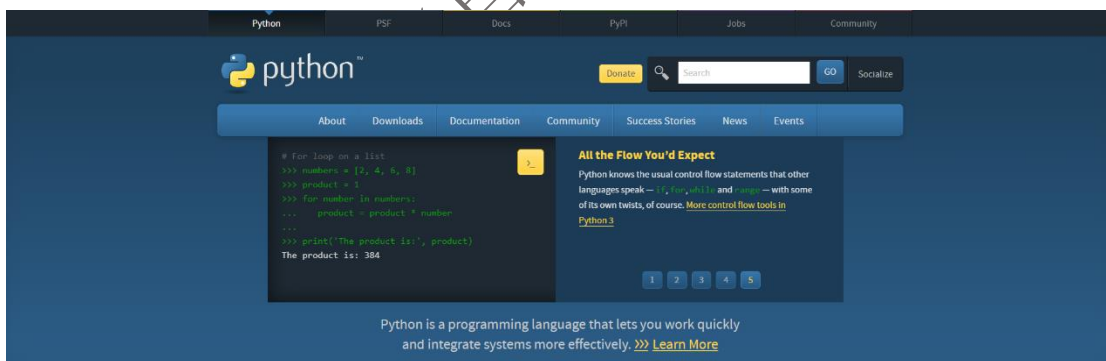
# 开发环境搭建

## 开发环境的概念

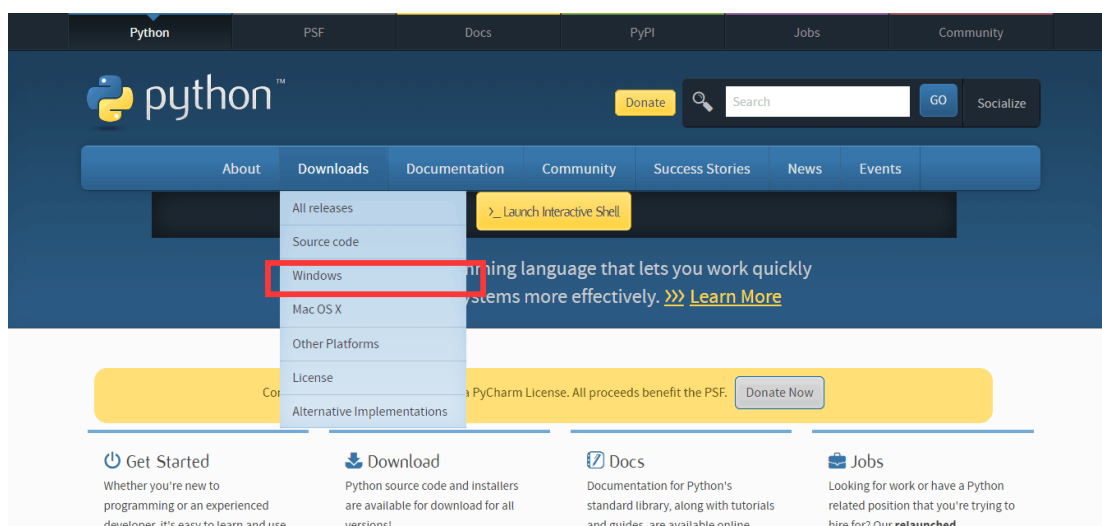
此处的开发环境指的是集成开发环境（IDE，Integrated Development Environment），是用于提供程序开发环境的应用程序，一般包括了代码编辑器、编译器、调试器和图形化用户界面工具。集成了代码编写、分析、[编译](#)、调试等功能的应用程序。下面介绍如何安装课程所需要用到的 IDE Visual Studio Code（VS Code）。

## 安装 Python

1. 打开浏览器输入网址并转到 <https://www.python.org/>，将会出现下图所示画面，若未跳转请检查网络连接。



2. 将鼠标移到 [Downloads](#) 后将会弹出下拉菜单，点击 [Windows](#)。



3. 点击 **Windows x86-64 executable installer** 开始下载。在浏览器左下角会显示下载进度。

## Python Releases for Windows

- [Latest Python 3 Release - Python 3.7.3](#)
- [Latest Python 2 Release - Python 2.7.16](#)

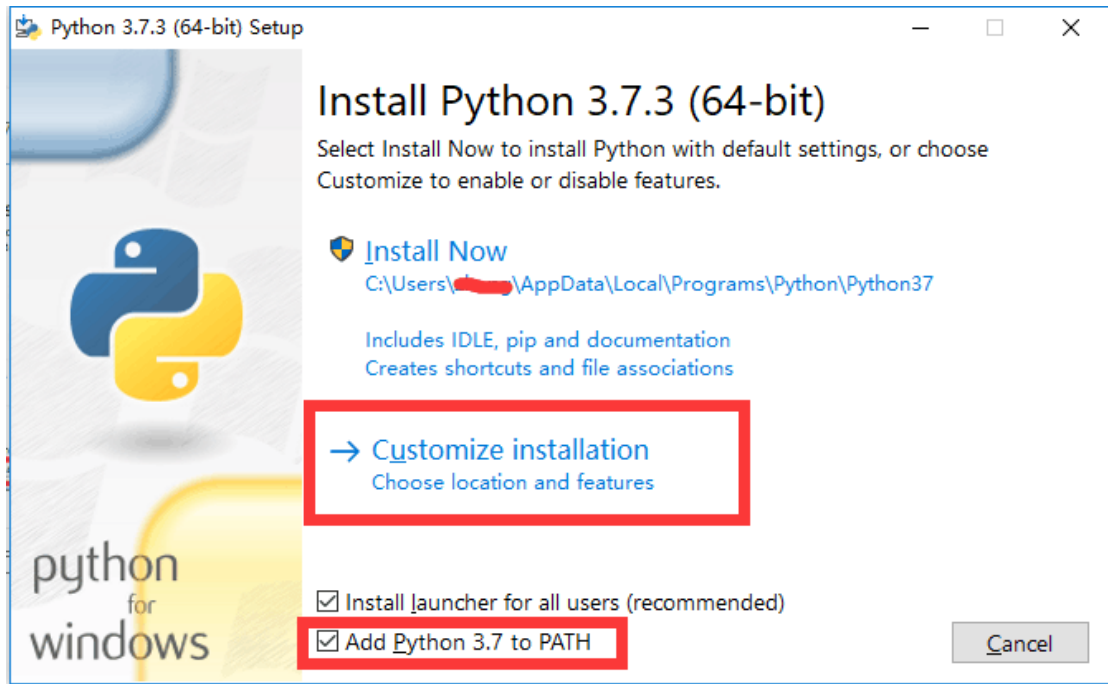
### Stable Releases

- [Python 3.7.3 - March 25, 2019](#)

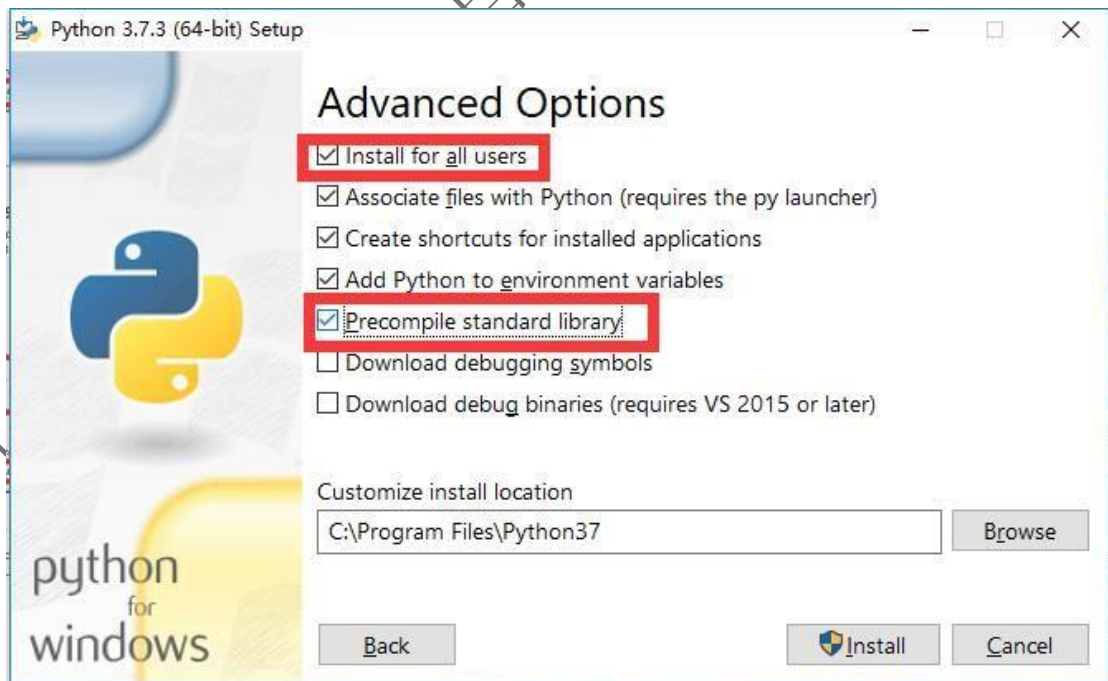
**Note that Python 3.7.3 cannot be used on Windows XP or earlier.**

- Download [Windows help file](#)
- Download [Windows x86-64 embeddable zip file](#)
- Download [Windows x86-64 executable installer](#)
- Download [Windows x86-64 web-based installer](#)
- Download [Windows x86 embeddable zip file](#)
- Download [Windows x86 executable installer](#)
- Download [Windows x86 web-based installer](#)

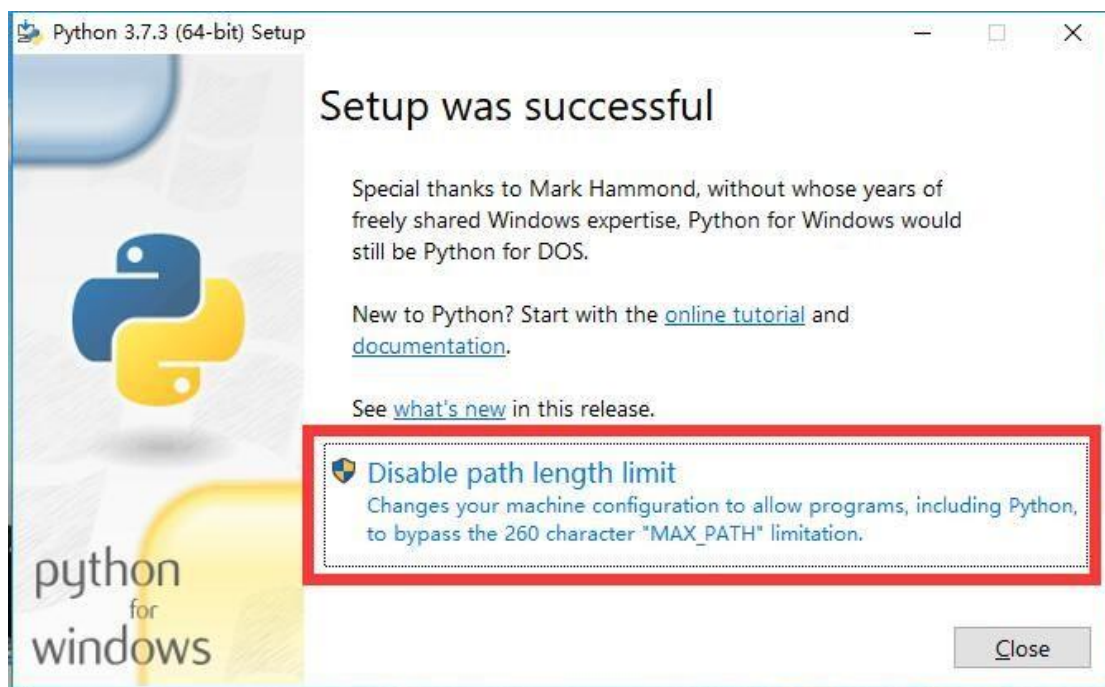
4. 下载完成后打开 [python-3.7.X-amd64.exe](#)。勾选 [Add Python 3.7 to PATH](#) 将 python 添加到系统环境变量中，再点击 [Customize installation](#)。



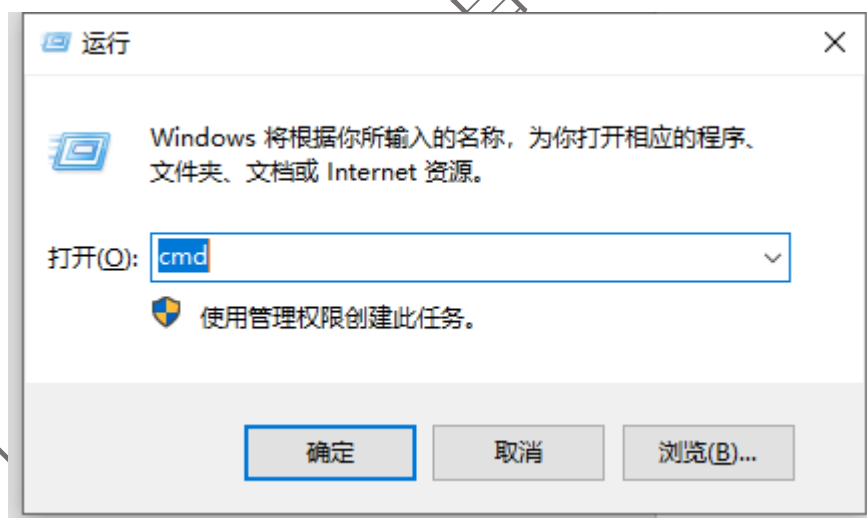
5. 勾选 *Install for all users* 和 *Precompile standard library* 后点击 *Install*。



6. 安装完成后点击 *Disable path length limit*。



7. 用键盘打出 **Windows + R** 的组合键后弹出 **运行** 窗口。输入 **cmd** 后点击 **确定** 弹出下图所示 **命令提示符**。





```
管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.17763.503]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()

C:\Users\Administrator>pip install python-ev3dev2
Collecting python-ev3dev2
  Using cached https://files.pythonhosted.org/packages/dd/af/b22f4f4b4e1cb98a1330f8ee9765d987fe571705e7cee4f4e256eeb3f96d/python-ev3dev2-2.0.0b3.tar.gz
Requirement already satisfied: Pillow in c:\program files\python37\lib\site-packages (from python-ev3dev2) (6.0.0)
Installing collected packages: python-ev3dev2
  Running setup.py install for python-ev3dev2 ... done
Successfully installed python-ev3dev2-2.0.0b3

C:\Users\Administrator>_
```

如果出现下图所示报错，请输入 `pip install python-ev3dev2`  
`--user` 后 回车。

```
C:\Users\...>pip install python-ev3dev2
Collecting python-ev3dev2
  Downloading https://files.pythonhosted.org/packages/dd/af/b22f4f4b4e1cb98a1330f8ee9765d987fe571705e7cee4f4e256eeb3f96d/python-ev3dev2-2.0.0b3.tar.gz (64kB)
    100% |#####| 71kB 71kB/s
Collecting Pillow (from python-ev3dev2)
  Downloading https://files.pythonhosted.org/packages/40/f2/a424d4d5dd6aa8c26636969decbb3da1c01286d344e71429b1d648bccb64/Pillow-6.0.0-cp37-cp37m-win_amd64.whl (2.0MB)
    100% |#####| 2.0MB 22kB/s
Installing collected packages: Pillow, python-ev3dev2
Could not install packages due to an EnvironmentError: [WinError 5] 拒绝访问: 'c:\\program files\\python37\\Lib\\site-packages\\PIL'
Consider using the '--user' option or check the permissions.

You are using pip version 19.0.3, however version 19.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

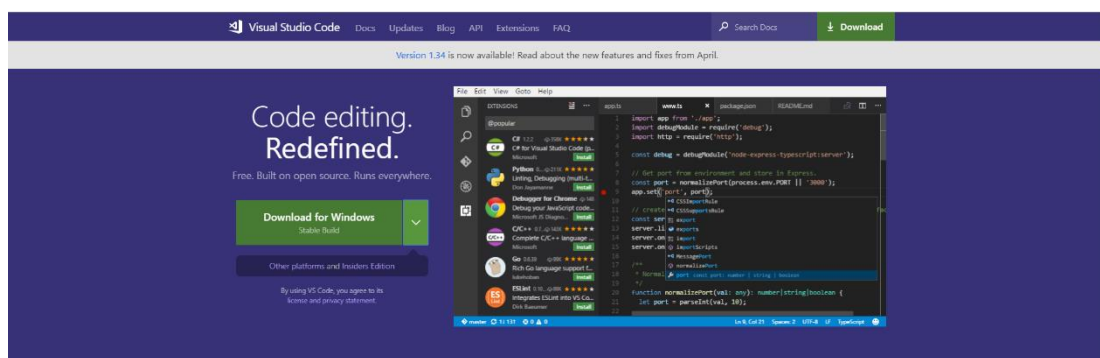
C:\Users\...>pip install python-ev3dev2 --user
Collecting python-ev3dev2
  Using cached https://files.pythonhosted.org/packages/dd/af/b22f4f4b4e1cb98a1330f8ee9765d987fe571705e7cee4f4e256eeb3f96d/python-ev3dev2-2.0.0b3.tar.gz
Collecting Pillow (from python-ev3dev2)
  Using cached https://files.pythonhosted.org/packages/40/f2/a424d4d5dd6aa8c26636969decbb3da1c01286d344e71429b1d648bccb64/Pillow-6.0.0-cp37-cp37m-win_amd64.whl
Installing collected packages: Pillow, python-ev3dev2
  Running setup.py install for python-ev3dev2 ... done
Successfully installed Pillow-6.0.0 python-ev3dev2-2.0.0b3
You are using pip version 19.0.3, however version 19.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

10. 进行到这一步恭喜你已经完成了 Python 的安装和环境配置以及库安装。

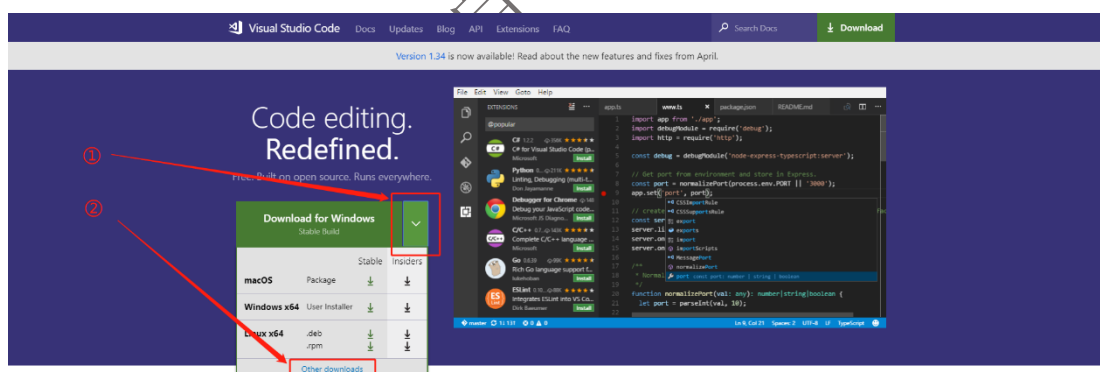


## 安装 Visual Studio Code

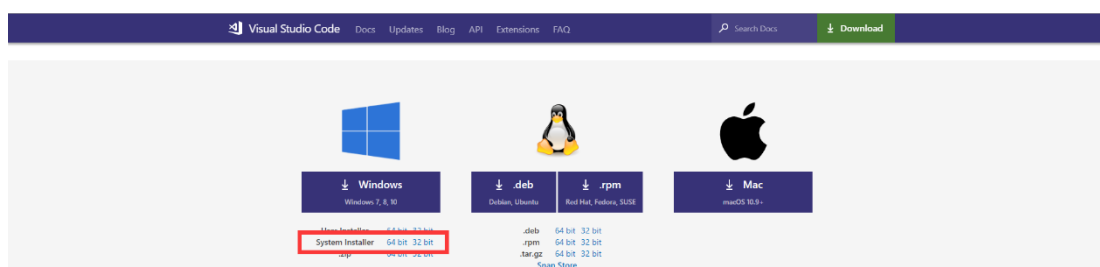
1. 打开浏览器输入并转到 <https://code.visualstudio.com/>，将会出现下图所示画面，若未跳转请检查网络连接。



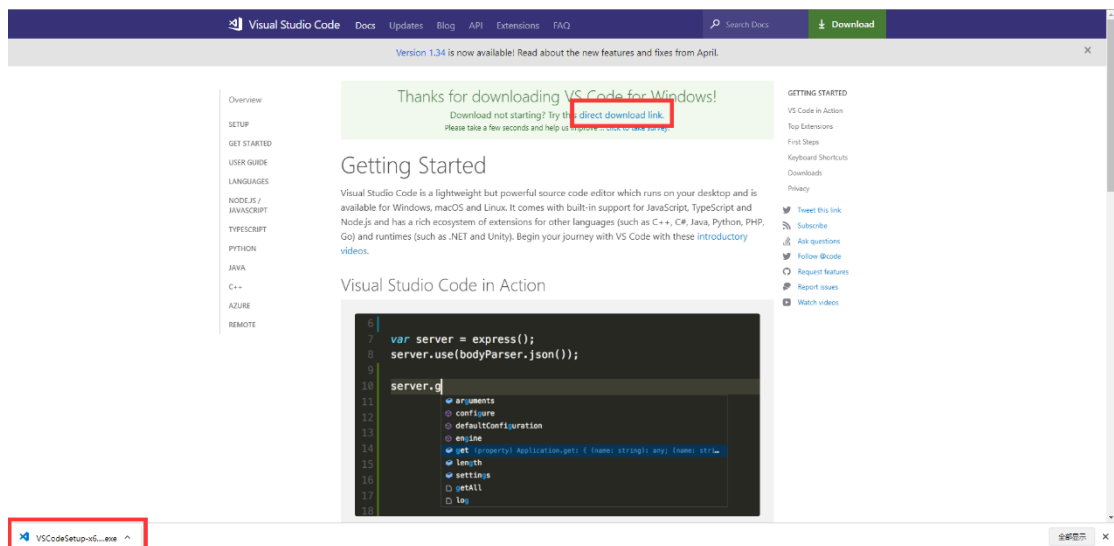
2. 点击 **Download for Windows** 旁边的下拉菜单按钮并点击 **Other downloads**。



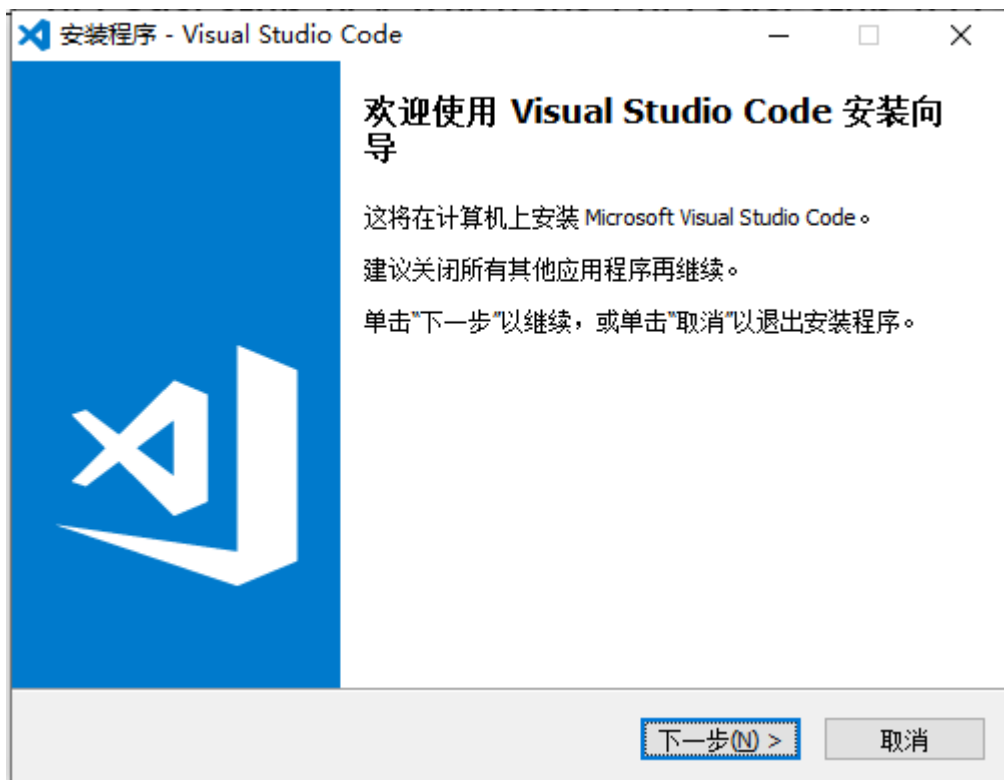
3. 转到下图所示界面，点击 **System Installer 64 bit 32 bit**。



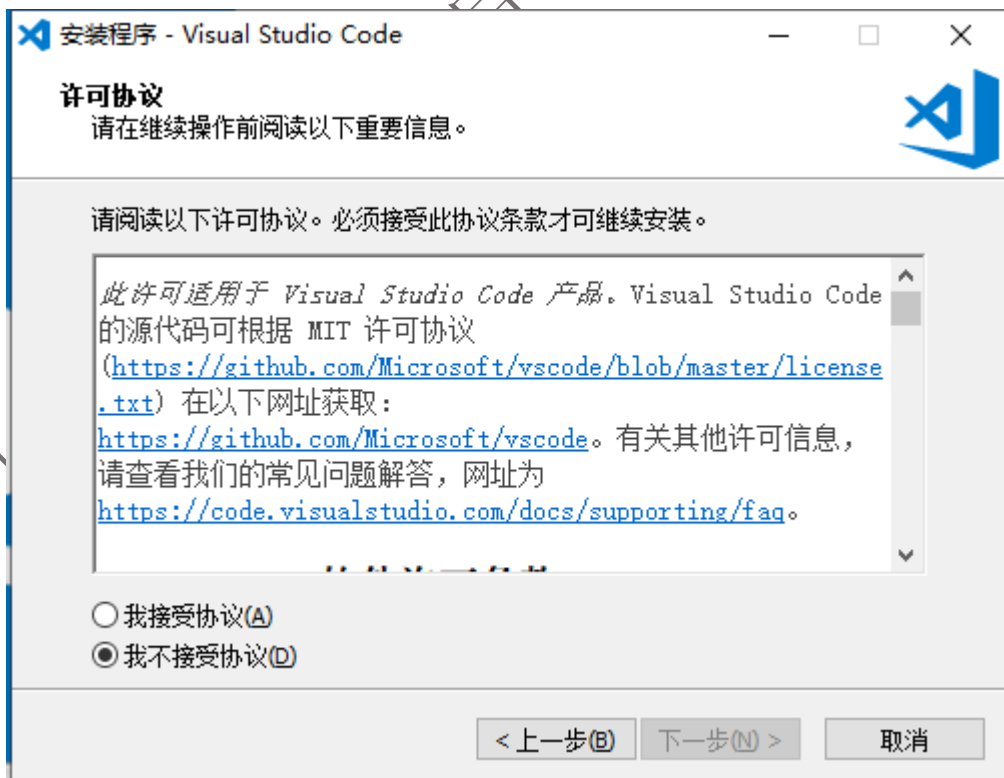
4. 点击 **System Installer 64bit** 后会跳转到如下页面，此时会显示类似左下角的下载状态，如没有开始下载，请点击 **direct download link**。



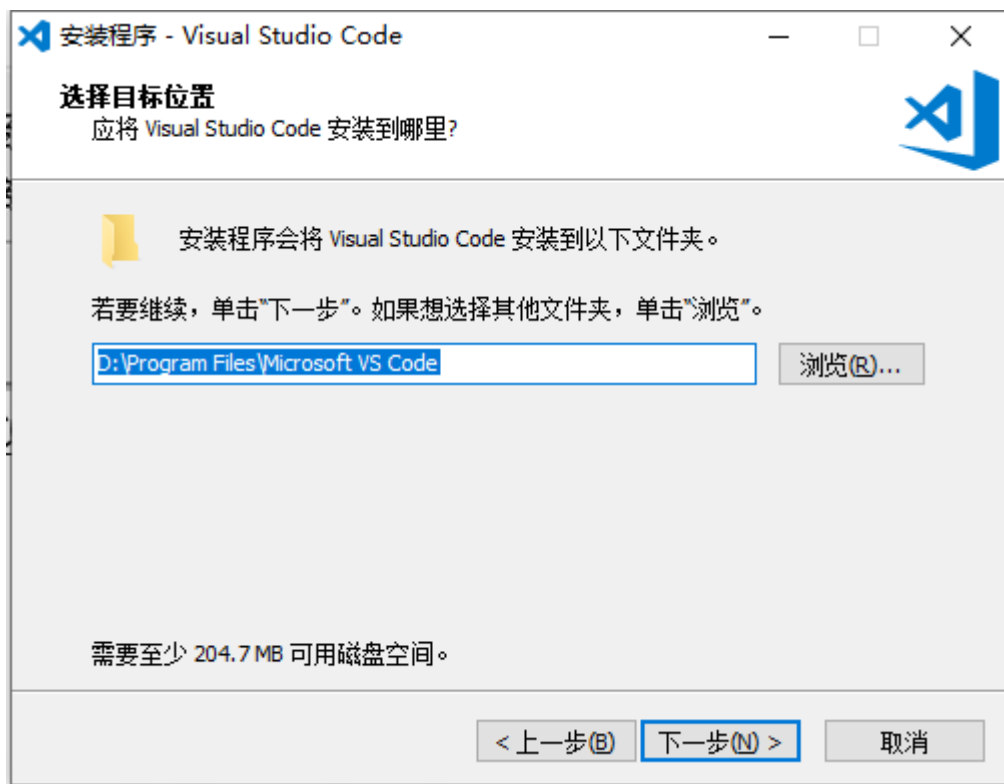
5. 下载完成后打开 **VSCodeSetup-x64-x.xx.x.exe** 并点击 **下一步**。



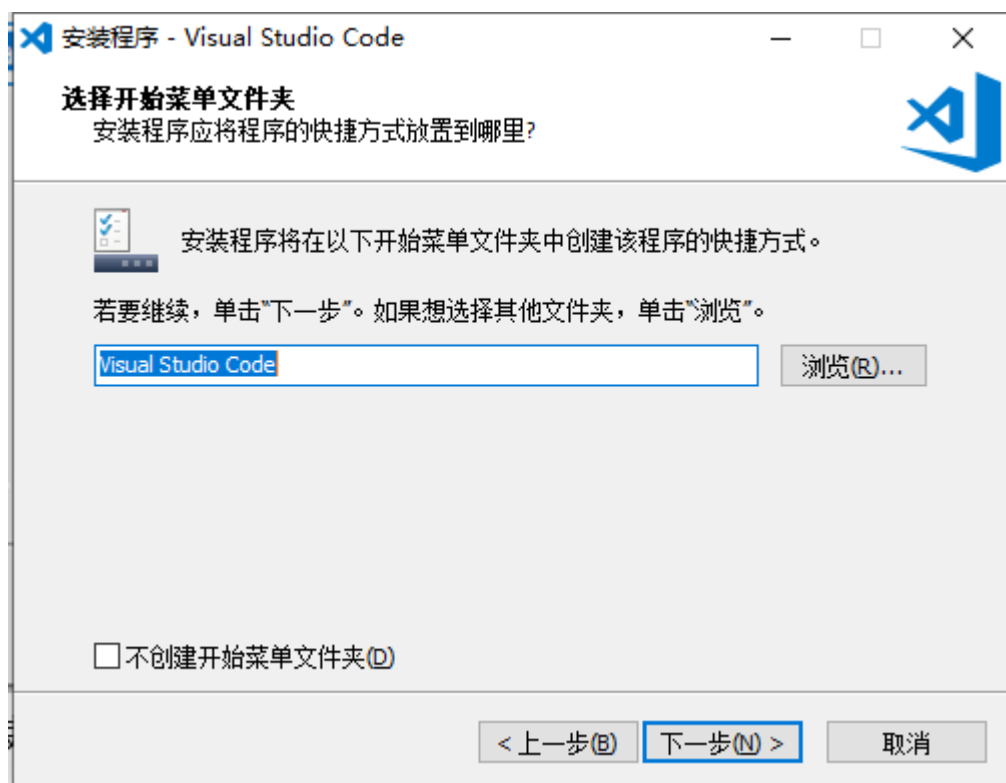
6. 点击 我接受协议 后点击 下一步。



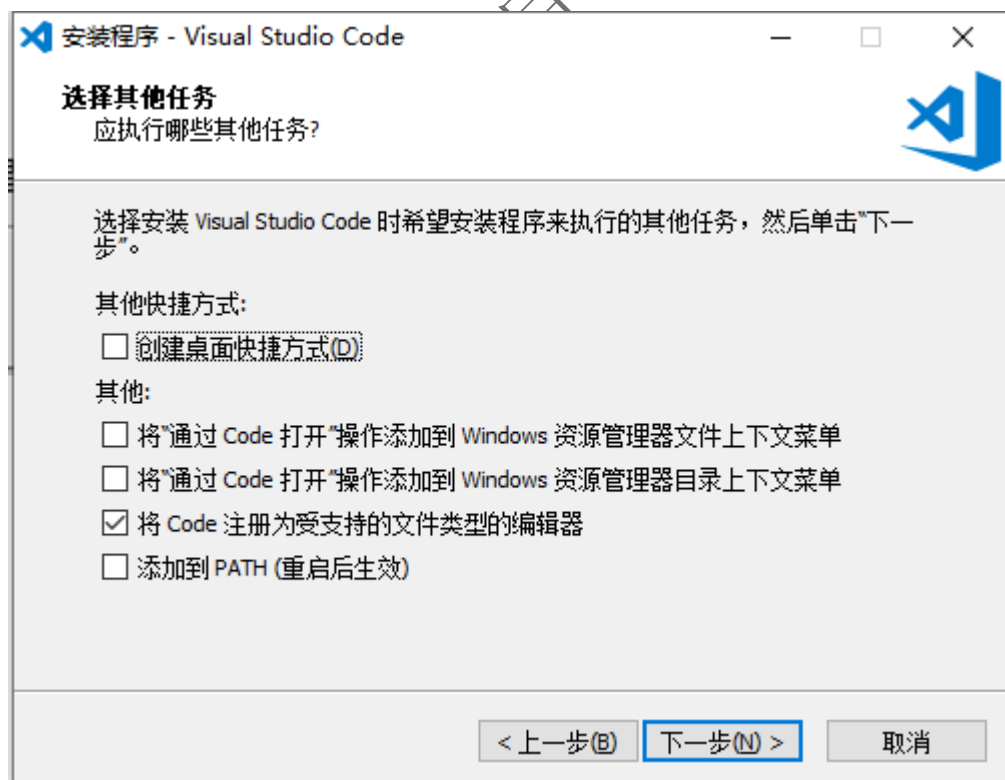
7. 根据自身电脑使用情况更改安装到文件夹的目录，然后点击 下一步。



8. 点击 下一步。



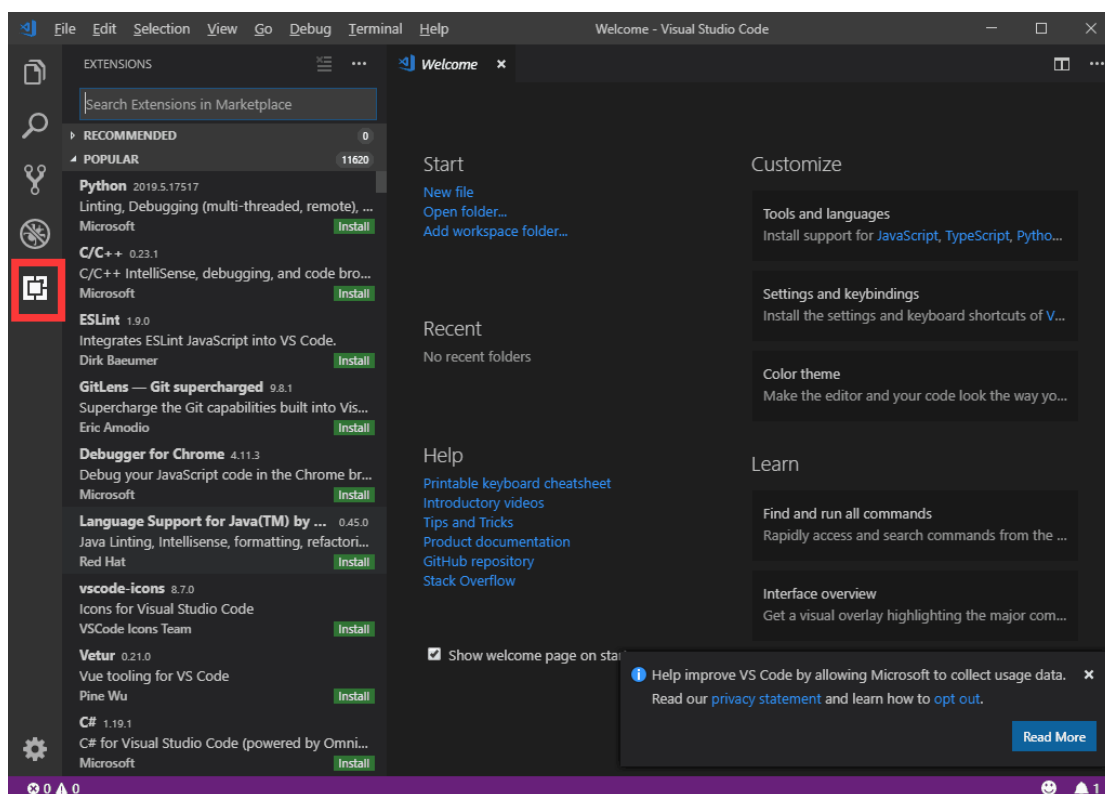
9. 点击 创建桌面快捷方式 后点击 下一步。



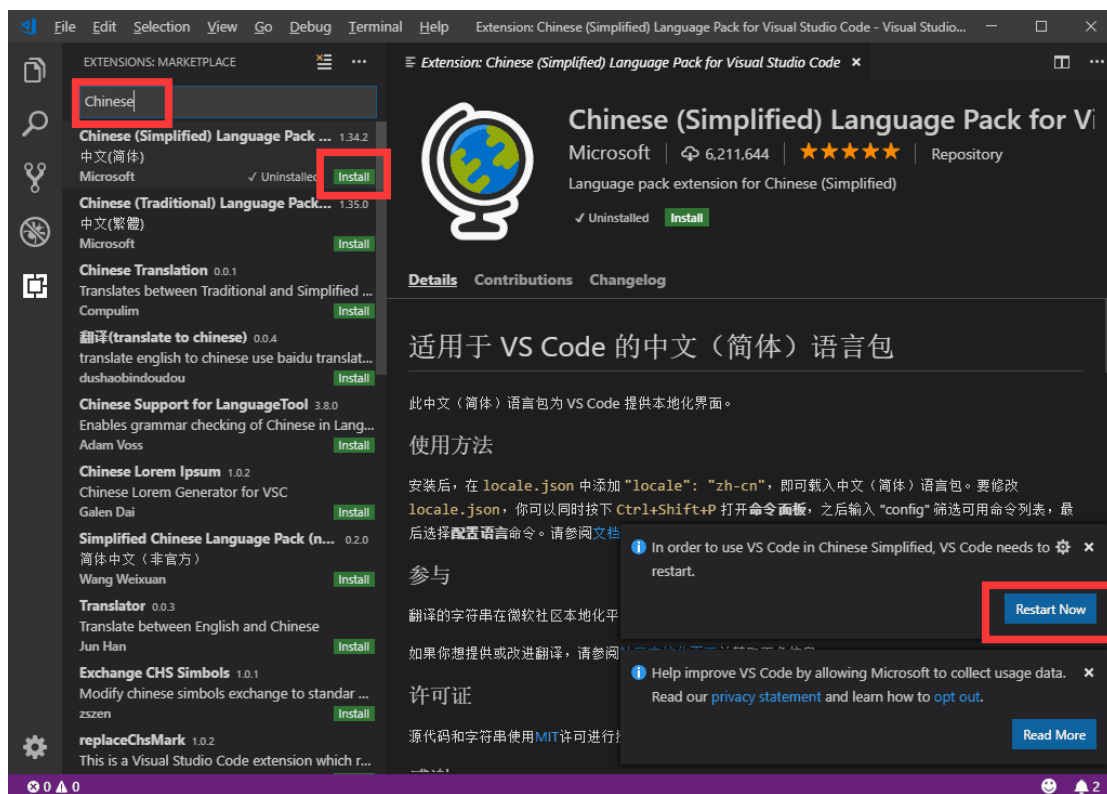
10. 点击 安装。弹出完成安装画面后点击 完成。



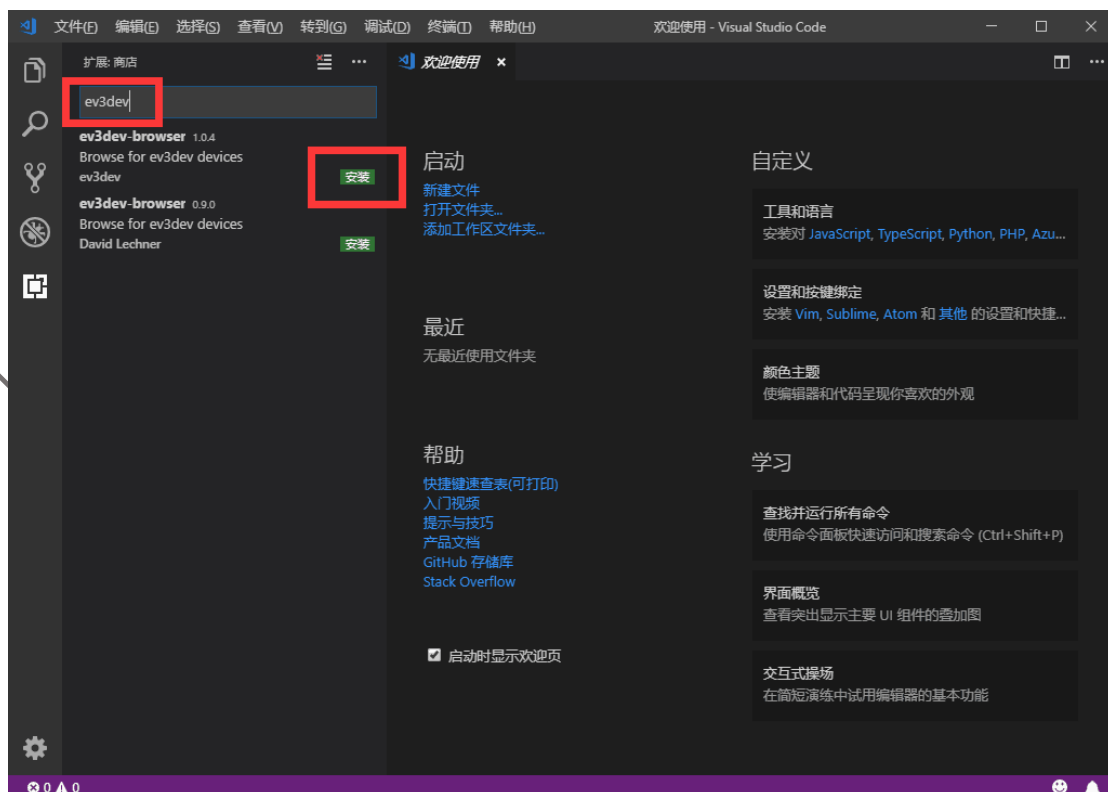
11. 启动 VSCode 后，点击左侧 拓展 (EXTENSIONS)。



12. 输入 Chinese 后点击 **Chinese (Simplified) Language Pack for Visual Studio Code** 右下角的绿色 **Install**，等待安装完成后下角会弹出如下所示弹窗，点击 **Restart Now**。



13. 重启 VSCode 后，再次点击左侧 拓展。输入 ev3dev 后点击 安装。





- 
14. 进行到这一步恭喜你已经完成了VSCode的安装和语言配置以及插件安装。

上海师范大学—“自动驾驶”小车项目组

---

# Python 基础

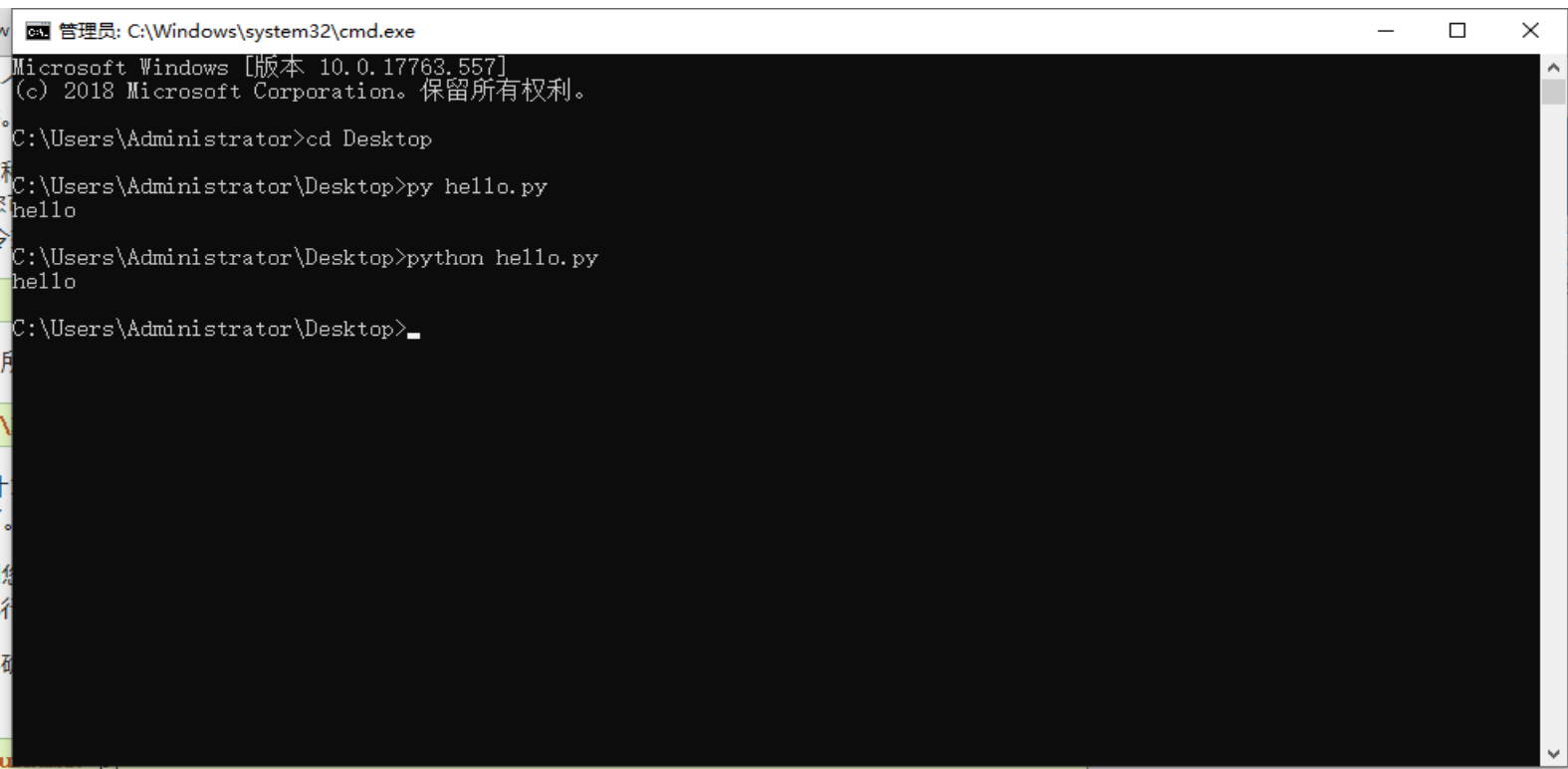
Python, 是一种面向对象的解释型计算机程序设计语言, 由荷兰人 Guido van Rossum 于 1989 年发明。Python 语法简洁清晰, 特色之一是强制用空白符 (white space) 作为语句缩进。Python 具有丰富和强大的库。它常被昵称为胶水语言, 能够把用其他语言制作的各种模块 (尤其是 C/C++) 很轻松地联结在一起。2017 年 7 月 20 日, IEEE 发布 2017 年编程语言排行榜: Python 高居首位。2018 年 3 月, 该语言作者在邮件列表上宣布 Python 2.7 将于 2020 年 1 月 1 日终止支持。用户如果想要在这个日期之后继续得到与 Python 2.7 有关的支持, 则需要付费给商业供应商。因此我们选择了学习 Python 3。

## 解释器

Python 是一门解释型语言, Python 规定了一套 Python 语法规则, 实现了 Python 语法的解释程序就成为了 Python 的解释器, Python 程序必须由 Python 解释器来处理运行。

调用 Python 解释器

在 window 系统中，可以通过 cmd（命令提示符）运行 Python 程序。通过键入 *python xxx.py* 或者 *py xxx.py* 来运行程序。



```
管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.17763.557]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>cd Desktop
C:\Users\Administrator\Desktop>py hello.py
hello
C:\Users\Administrator\Desktop>python hello.py
hello
C:\Users\Administrator\Desktop>_
```

除了直接运行写好的程序外还可以通过进入 Python 解释器的交互模式实时执行，评估 Python 语句或表达式。通过在 cmd 中键入 *python* 或者 *py* 进入交互模式，在进入交互模式后通过 *exit()* 或者 *Ctrl+Z* 退出交互模式。

## 解释器的运行环境

默认情况下，Python 的源码文件以 UTF-8 编码形式处理。在这种编码方式中，世界上大多数语言的字符都可以同时用于字符串字面值、变量或函数名称以及注释中——尽管标准库中只用常规的 ASCII 字符作为变量或函数名，而且任何可移植的代码都应该遵守此约定。

---

要正确显示这些字符，你的编辑器必须能识别 UTF-8 编码，而且必须使用能支持打开的文件中所有字符的字体。

如果不使用默认编码，要声明文件所使用的编码，文件的第一行要写成特殊的注释。语法如下所示：

```
# -*- coding: encoding -*-
```

其中 encoding 可以是 Python 支持的任意一种 codecs。

关于第一行规则的一种例外情况是当源代码在 Linux/Unix 系统运行时，编码声明就要写在文件的第二行。这种例外情况会一直出现在我们的课程中。

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
```

基本数据类型

运算符

数字

字符串

流程控制

管理员: C:\Windows\system32\cmd.exe

Microsoft Windows [版本 10.0.17763.557]  
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>cd Desktop

C:\Users\Administrator\Desktop>py hello.py  
hello

C:\Users\Administrator\Desktop>python hello.py  
hello

C:\Users\Administrator\Desktop>python  
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("hello")  
hello  
>>> exit()

C:\Users\Administrator\Desktop>py  
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> ^Z

C:\Users\Administrator\Desktop>\_

“ ”  
上海师范大学

---

# 运动控制项目书

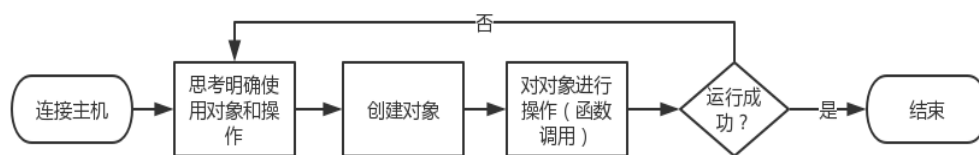
运动控制，就是通过编写程序使小车动起来。小车的运动可以分为直线运动和曲线运动，小车运动即车轮转速 $\neq 0$ 。直线运动即车轮转速 $\neq 0$ 且两轮的转速和方向相同，曲线运动即车速 $\neq 0$ 且两轮转速不一致，存在转速差。

## 目标

1. 体会控制车轮与控制小车的不同
2. 小车直线运动到指定位置
3. 小车转弯到指定位置
4. 小车走出正方形的路径

## 构建

本章节是项目的第一章节，重点在于掌握 ev3dev2 库的 Motor 类和 on 类函数的搭配使用。面向机器人编程与面向对象编程在本项目中是一样的。首先建立电脑与 EV3 主机的连接，连接主机后不是急着敲代码，而是在于先动脑思考程序逻辑，明确编写程序用到的对象（类）和对象执行的操作（函数调用）。



以下即为本章主要内容：

功能	涉及知识点	预期目标
直线行驶	(1) On 类、Tank 类 控制语句 (2) 赋值	行驶预定时间或预定距离
弯道行驶 实现正方形	(1) On 类、Tank 类 控制语句 (2) 赋值 (3) 循环语句	(1) 精确转动指定度数 (2) 使小车运动到某一位置

## 知识铺垫

在开始本堂课的项目实践之前，你需要掌握以下知识点：

### 1.物理知识：

- (1) 调参
- (2) 小车转向原理

### 2.数学知识：

- (1) 赋值的矢量型

### 3.编程知识：

---

(1) 控制语句

(2) 循环语句

### *On 类函数*

on 类函数是直接控制电机的函数，也是运动控制的核心函数，需要注意的是所有的 on 类函数都含有 speed 这个必填参数。

```
#!/usr/bin/env python3
# -*- encoding: utf-8 -*-

from ev3dev2.motor import *

#创建对象
__motor = Motor(OUTPUT_A)

while True:

    #调用函数，电机以大约最大速度的 30%旋转
    __motor.on(speed=30)

    Print(__motor.speed())
```

### *单电机控制程序 Demo*

对于机器人小车而言，大型电机多数情况下成对出现。针对这种成对的电机组，延伸出了新的类—MoveTank 类。MoveTank 类可以



---

调用部分 on 类函数，它可以同步控制一组电机，从而减少了单独控制的延时。请同学们仿照上述单电机控制程序写出控制电机组的程序完成目标 2 和 3。

实际操作时效果和理想效果存在偏差。这就需要我们不断测试，直到达到最佳效果。这种根据实际情况不断进行测试寻找参数值，得出理想结果的过程叫调参。

在着手目标 4 之前，请同学们分析正方形的特性，结合 Python 语法，使用流程控制语句进行编程。

## 初识传感项目书

上一章节运动控制项目中，我们在调参实现走出正方形的过程中会出现一些意料之外的情况，这时没有一个开关中断，也没有避障的功能，就像我们骑自行车失控之后没有刹车只能脚刹一样。本章节将介绍超声波传感器和触动传感器让小车智能起来。

### 目标

1. 了解触动传感器与超声波传感器的原理与功能

2. 通过触动传感器控制小车运动与停止
3. 运用超声波传感器测距与控制车速
4. 注意传感器优先级在程序中的体现
5. 综合实现上述 4 个目标

## 构建

功能	涉及知识点	预期目标
控制小车运动状态	(1) 触动传感器原理 (2) While 循环语句 (3) If 判断语句 (4) break 与 continue 功能	触动传感器直接实现小车运动与停止
避障	(1) 超声波传感器原理 (2) 定义函数 def (3) While 循环语句 (4) If 判断语句 (5) 赋值	(1) 小车行驶至安全距离停止 (2) 车速随与障碍物的距离而改变

## 一、知识铺垫

在本堂课开始之前，你需要掌握以下知识点：

1.物理知识：

- (1) 调参
- (2) 超声波原理

2.数学知识：

- (1) 正比例函数

3.编程知识：

- (1) python 基础

## 二、实现方法

### 1) 控制小车运动状态

触动传感器的使用方式有三种：(1) 等待按下传感器；(2) 等待触感器被释放；(3) 等待按下传感器，然后松开。另外，is\_pressed 能够指示当前触摸传感器是否被按下的布尔值（即 True 或 False）。流程图 2.1 实现小车点火与熄火的功能：

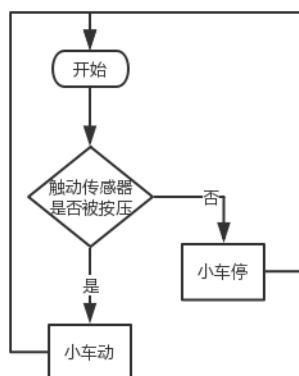


图 2.1

这种做法需要我们一直跟着小车，按住小车上的触动传感器才能保证小车前进，下面我们给出优化的流程图 1.2:

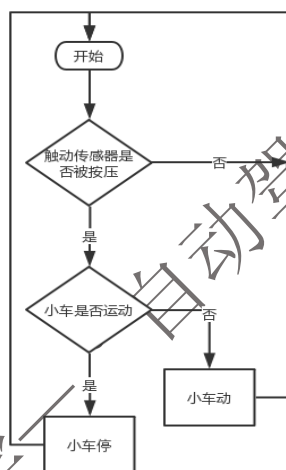


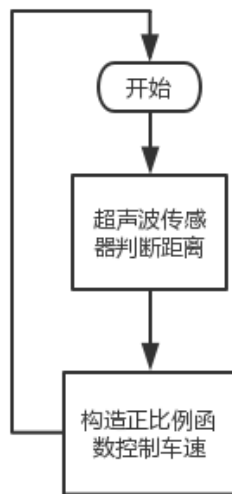
图 2.2

## 2) 避障

本节课中通过超声波传感器得到小车与障碍物的距离来实现避障。我们只需要利用 `debug_print` 函数就能够将数值打印在我们的电脑屏幕上了。

一般情况下，我们提前设定一个安全距离，小车行驶至该安全距离时停止。

现在讨论车速控制问题，为了实现小车越靠近障碍物车速越慢的情况，可以利用正比例函数，以小车与安全距离的差值为自变量，车速为因变量，系数决定了车速的变化程度。



我们给出流程图 2.3:

图 2.3

### 3) 考虑传感器的优先级

考虑如下情况：小车行驶至安全距离时，超声波传感器将控制小车停止，此时触动传感器并没有被通过按压控制小车停止，那么两个传感器所期望的小车接下去的行为就会相互矛盾。

两个传感器的优先级取决于小车是否行驶至安全距离：若已到达，超声波传感器优先级较高；若未到达，触动传感器优先级较高。优先级的体现将会在代码中以 if 判断语句的先后顺序体现。

### 4) 总流程图

本节课的最后我们要实现通过触动传感器让小车前进，小车在距离障碍物一定距离时停止，且小车速度随障碍物的远近改变。下面给出本节课的总流程图 2.4:

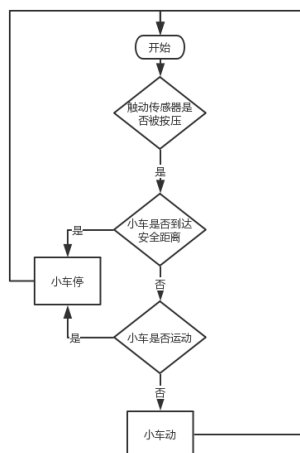


图 2.4

# 光电循迹项目书

在运用了触动和超声波传感器之后我们确保了小车行驶的安全。  
在现实中车辆总是沿着线行驶的，但是我们的小车是无法和线产生关联的，于是我们就要用到颜色传感器通过反射光的反馈值来调节我们的小车状态。

## 目标

- 1. 了解颜色传感器的几种用法
- 2. 通过颜色传感器反馈值的变化改变两轮的速度
- 3. 让小车能够循线进行圆周运动
- 4. 将目标 3 融入到初识传感的程序中，优先级从高到低的顺序为，  
触动传感器 > 超声波传感器 > 颜色传感器

## 构建

功能	涉及知识点	预期目标
颜色设定值确定	(1) 光学 (2) 传感器原理	使设定值能够作为巡线的标准

速度赋值	(1) 相对速度 (2) 速度赋值语句	(1) 能够运用赋值语句对小车速度进行赋值 (2) 理解左转和右转对速度赋值的差别
巡线		(1) 左右轮速度能够根据返回值和设定值改变 (2) 确定小车运动方向和内外圈，对小车左右轮速度进行合理赋值

## 一、

在本堂课开始之前，你需要掌握以下知识点：

### 1. 物理知识：

- (1) 光学
- (2) 相对运动

### 2. 编程知识：

- (1) python 基础

## 二、实现方法

### 1. 速度赋值

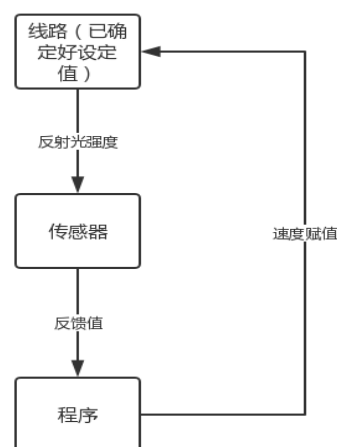
---

我们现在又有了新的问题，如何去运用它呢？它在巡线中到底起着什么样的作用？下面将详细讲述传感器和校车运动的联系。

(1) 背景知识：tank 函数进行速度赋值

(2) 1. 设定值的确定：巡线巡的是线路的边缘，即颜色传感器在小车运动过程中，始终在线路边缘的上方。所以首先测定线路上颜色传感器的返回值以及地板颜色传感器返回值，然后两者相加除以二，即为线路边缘的颜色返回值。

2. 流程图



## 2. 圆周运动巡线

现在的我们已经让小车能动起来了，并通过传感器能走直线，那么拐弯的话，我们该如何做呢？

(1) 背景知识：相对运动（第一节内容）

(2) 巡线过程：

(a) 顺时针、沿外圈:

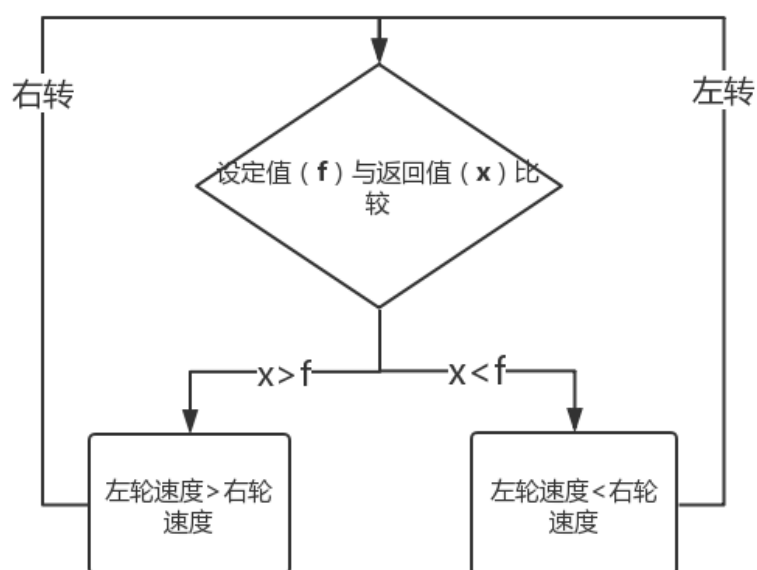


图 (a)

(b) 顺时针、沿内圈:

$x < f$ : 左轮速度 < 右轮速度, 右转

$x > f$ : 左轮速度 > 右轮速度, 左转

**注意:** 若小车速度过快, 在右转时容易冲出跑道。

**解决方法:** 1. 小车速度减小

2. 线路外圈半径增大

(c) 逆时针、沿外圈:

与 (b) 相同

(d) 逆时针、沿内圈:

与 (a) 相同

(e) 倒车巡线 (不做重点):



---

1. 传感器问题：倒车的时候前置传感器已经不能使用，因为这对倒车时处于前方的小车尾部有延迟，无法精确巡线。所以需要在尾部另加传感器。

2. 巡线转弯（以顺时针、沿外圈为例）：

$x < f$ ：左轮速度 < 右轮速度，右转

$x > f$ ：左轮速度 > 右轮速度，左转

### 3. 结合初识传感

结合上节讨论的优先级问题，本节内容接着引入颜色传感器到优先级问题讨论中。

首先，我们假定现在小车循线行驶

然后，讨论两种情况：

（a）未到安全距离：

此时我们同时需要用触碰传感器和颜色传感器，通过触碰传感器控制小车运动与否，颜色传感器用来巡线。若让小车能在巡线的过程中停下，需要设置触碰传感器优先级最高，即

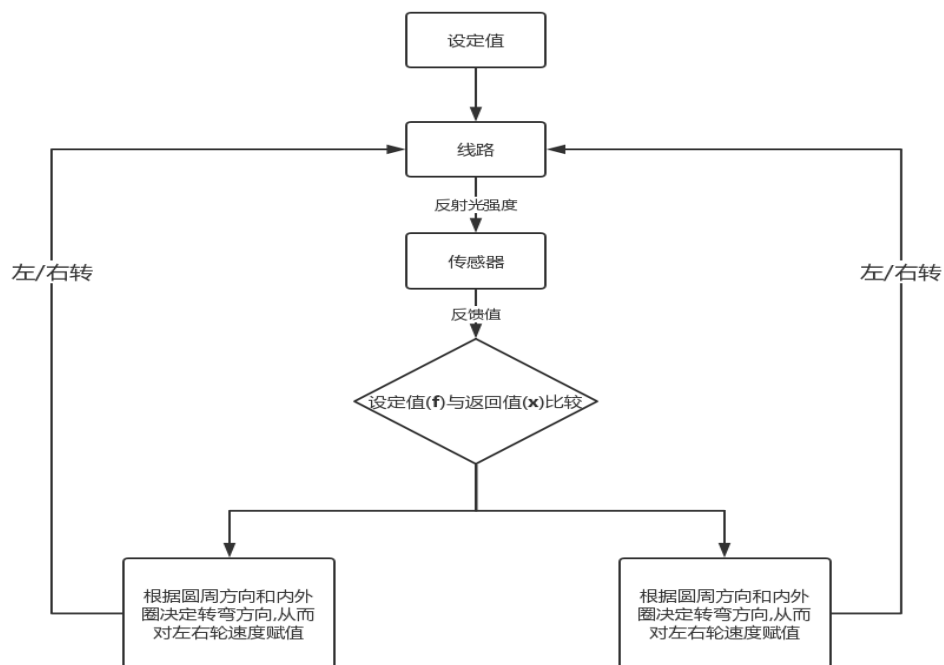
**触碰传感器 > 颜色传感器**

（b）到达安全距离内：

从上节内容我们已知超声波传感器优先级大于触碰传感器。那么，根据(a)中结论，易知

**超声波传感器 > 触碰传感器 > 颜色传感器**

### 三、总流程图



## 算法循迹项目书

上一章节我们实现小车循线做圆周运动，但是我们会感受到局限性，因为速度是随着颜色传感器的反馈值做跳变的缺少了智能控制的元素。本章节将通过算法实现小车智能运动在直线和弯道上达到自动

---

驾驶的效果。希望同学们结合攻略中的微积分和 PID 算法实现本章 w 目标。

## 目标

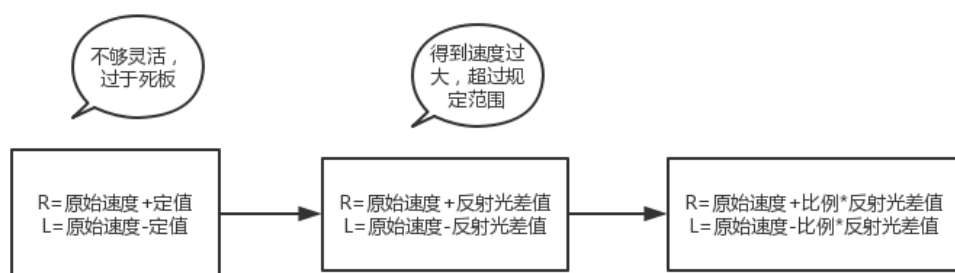
1. 了解比例和微积分在小车循线过程中的物理意义
2. 了解 PID 各种组合的控制对小车的影响
3. 熟悉 PID 调参
4. 小车循线
5. 将目标 4 融入到初识传感的程序中，优先级从高到低的顺序为，  
触动传感器 > 超声波传感器 > 颜色传感器

## 构建

目标一：

假设你现在需要将桶里的水接到指定位置，你需要每五分钟出来看一次，当水快接近指定线时，可将水阀适当调小，以使到达水位更加精准。这就和我们现在要解决的问题相类似。

观察上一章节程序代码可发现小车在巡线时两轮速度是固定的，这样不能达到小车随反射光值改变，但是如果直接输入反射光值改变又会过于大，这时我们需要对反射光值进行一个比例缩放。（详细知识见 P5）



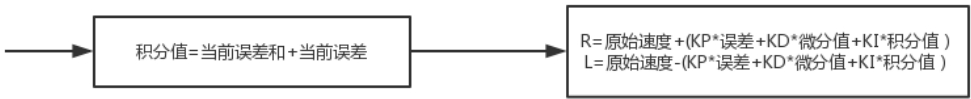
完成以上内容，小车可基本实现循迹直线行走，但在转弯时你会发现小车存在大幅度摆头，或者冲出弯道，这存在很大安全隐患问题。为解决这一问题，我们需要对当次的反射光差值与上一次的反射光差值进行做差。这种方法是我们的微分调节的一种简化。（微分知识详见P5）



现假设你每个月都必须剩且仅剩余 1000 元钱，因此你将自己每个月得到的钱和花出去的钱进行记录然后求和，支出为正，收入为负。这样就可以根据结果看出自己为了达到 1000 元的目标是应该增大收入还是减少支出。这和我们现在要解决的小车循迹的最后一个问题相类似。

完成以上两个内容后，差值也并不一定为 0，为了解决这一问题，我们需要将每次的误差按正负积累，那么即使系统在比例控制和微分控制部分已经趋于稳定，只要误差不为 0 就会存在静差，只要存在残余

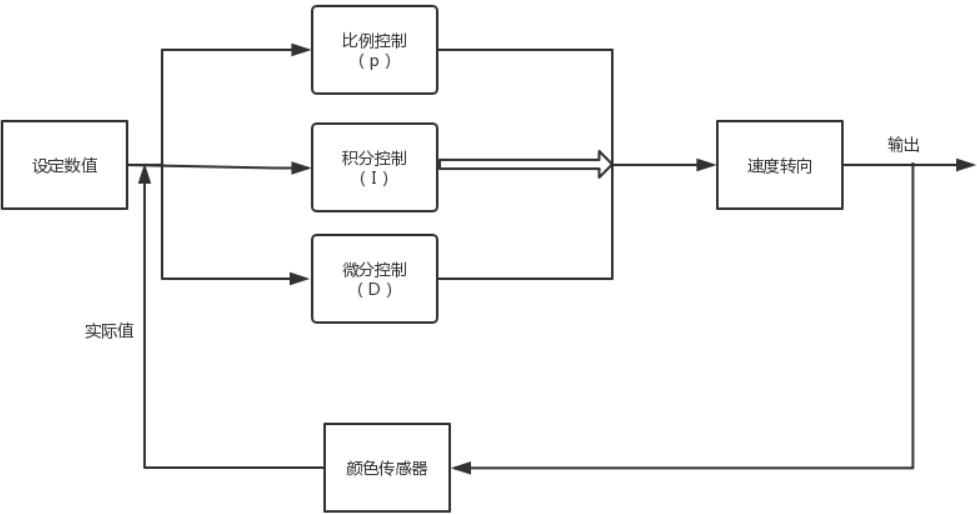
偏差那么积分就会对系统产生影响，直到系统的差值为 0。认真思考这一问题，你会发现这一问题和积分的几何意义相近，我们选用积分来解决这一问题（积分知识详见 P5）



目标二：

通过对 PID 算法的各种组合进行实践完成指定路线，观察完成效果，得出结论。（可参见 P5 进行比较验证）

目标三、四、五：

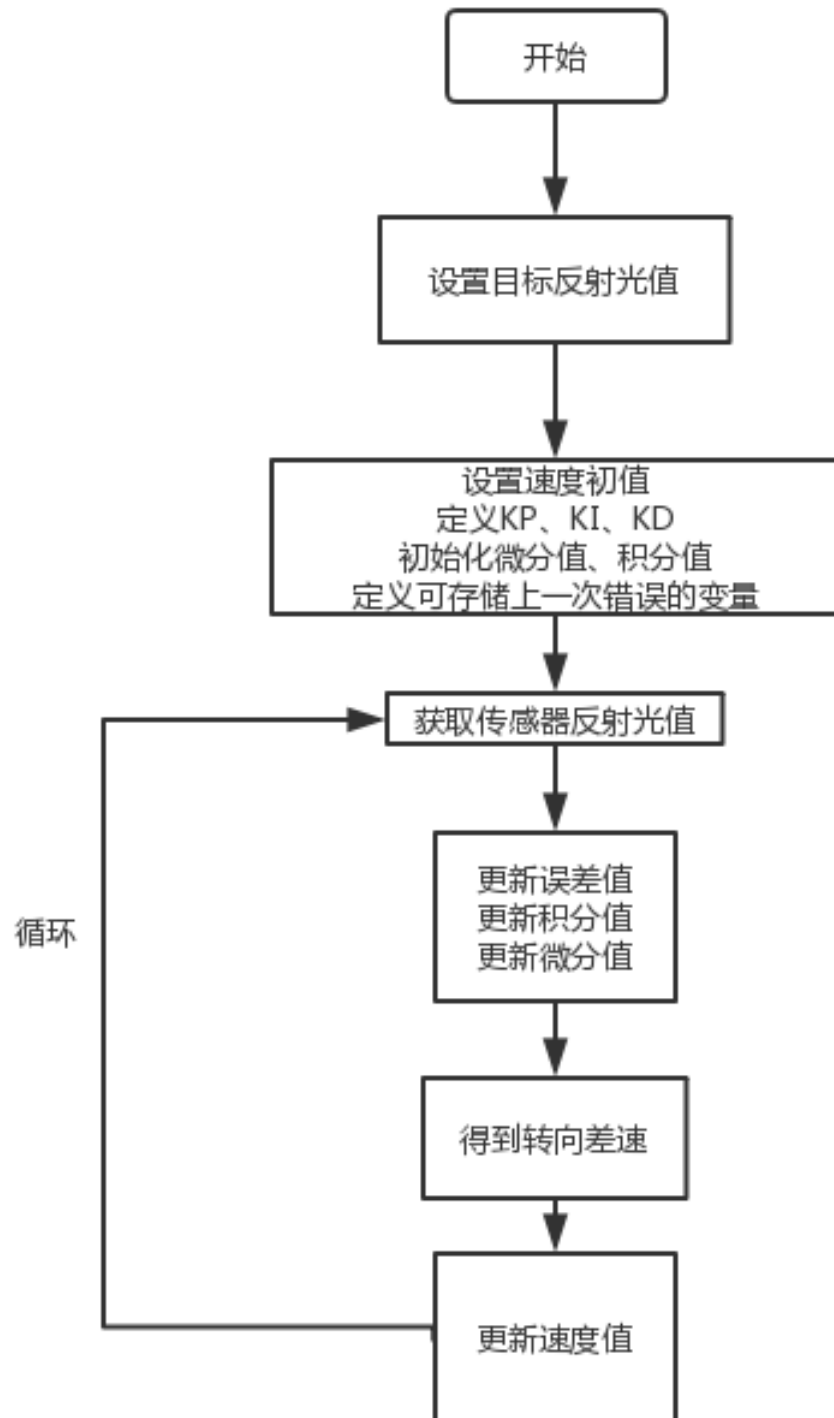


根据上述流程图可完成算法循迹代码，KP、KD、KI 的值需要根据实际线路进行参数调整，实现小车的循迹，按照该流程可以类比到触动传感器和超声波传感器，进行实验。

在转弯时遇到问题适当调节 D，直线摆头过大适当调节 P、消除残余

偏差调节I。

项目完整流程图：



# 识别目标项目书

经过前面的学习，我们熟悉并运用了颜色传感器控制小车自动驾驶，但是颜色传感器对于处理复杂的现实环境力不从心，抗干扰能力不强，本章节我们将介绍使用包含了颜色传感器功能的高阶传感器——摄像头（OpenMV）。通过了解控制摄像头（OpenMV），小车将对环境做出“应激性”。

## 目标

- 1. 了解图像处理的背景知识
- 2. 了解并进行摄像头初始化
- 3. 找到摄像头（OpenMV）画面中想提取的颜色
- 4. 寻找最大的色块

## 构建

在本章的摄像头传感器的学习中，我们将学习如何使用 OpenMV——一个对摄像头进行编程的软件，并通过 OpenMV 的一些基础操作完成识别目标的目的，以下即为本章主要内容：

功能	涉及知识点	预期目标
颜色阈值确定	(1) Python 语言基础 (2) OpenMV 基本使用方法	(1) 能够粗略获取一个图像中某颜色的颜色阈值；

	(3) OpenMV 阈值调节器	(2) 通过阈值处理器获取精确颜色阈值。
识别目标（框选目标色块）	(1) 寻找色块语句 (2) 画选取框语句 (3) 定义寻找最大色块函数 (4) 学会使用 Merge 参数	(1) 通过阈值编辑器设置目标颜色，框选目标颜色所处区域。 (2) 去除细小选取框，合并多个选取框

## 一、知识铺垫

在开始本堂课的项目实践之前，你需要掌握以下知识点：

### 1.物理知识：

- (1) 小孔成像原理
- (2) 帧率

### 2.数学知识：

- (1) 平面直角坐标系
- (2) 空间直角坐标系
- (3) LAB 色彩
- (4) 阈值的基本概念

### 3.编程知识：

- (1) Python 基础

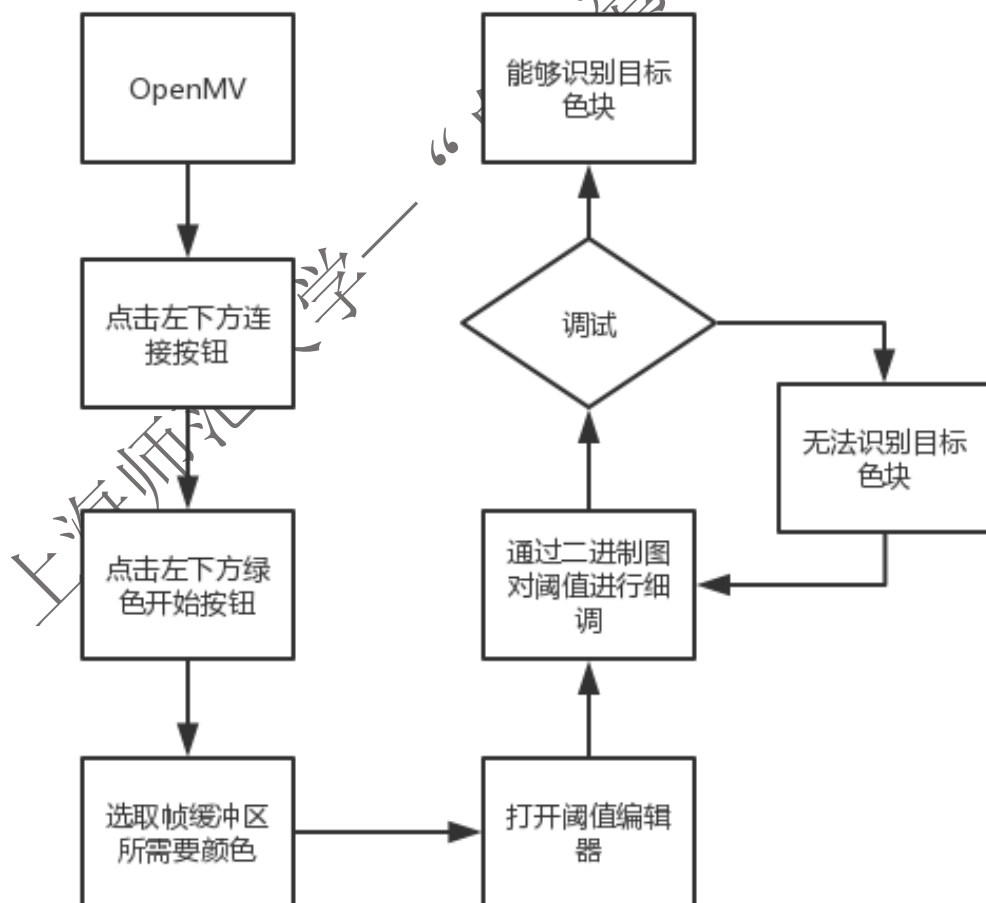


## 二、流程图

### 1、阈值确定：

新知识点学习：阈值的粗细调节

在获取颜色阈值之时，如果只用粗调，在外界条件变化时（如光照变化、视角远近调整），容易导致无法寻找到目标色块。而如果只使用阈值控制器进行调节，容易参杂其他颜色（虽然未显示在图像上，但在后续使用中将潜在影响摄像头使用）且调试难度较大。因此通过粗调确定颜色，细调框定范围能够更精确地获得所需颜色阈值。



---

## 2.识别目标

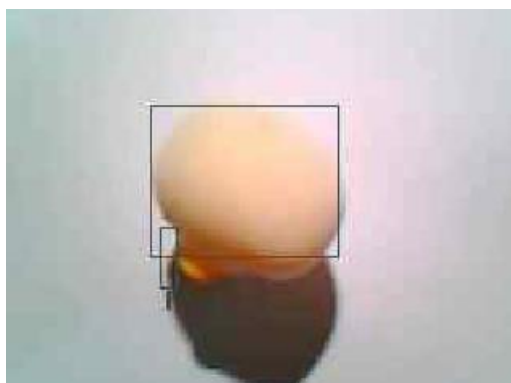
新知识点学习 (1) 寻找色块语句

(2) 画选取框语句

(3) 定义寻找最大色块函数

(4) 学会使用 Merge 参数

在只使用 `find_blobs([颜色阈值])` 语句会使得图片中显示多个小抓取框，如何使得摄像头将所有色块合并到一起呢？



方法一：（该方法适用于像素间隔较小的多个选取框合并）

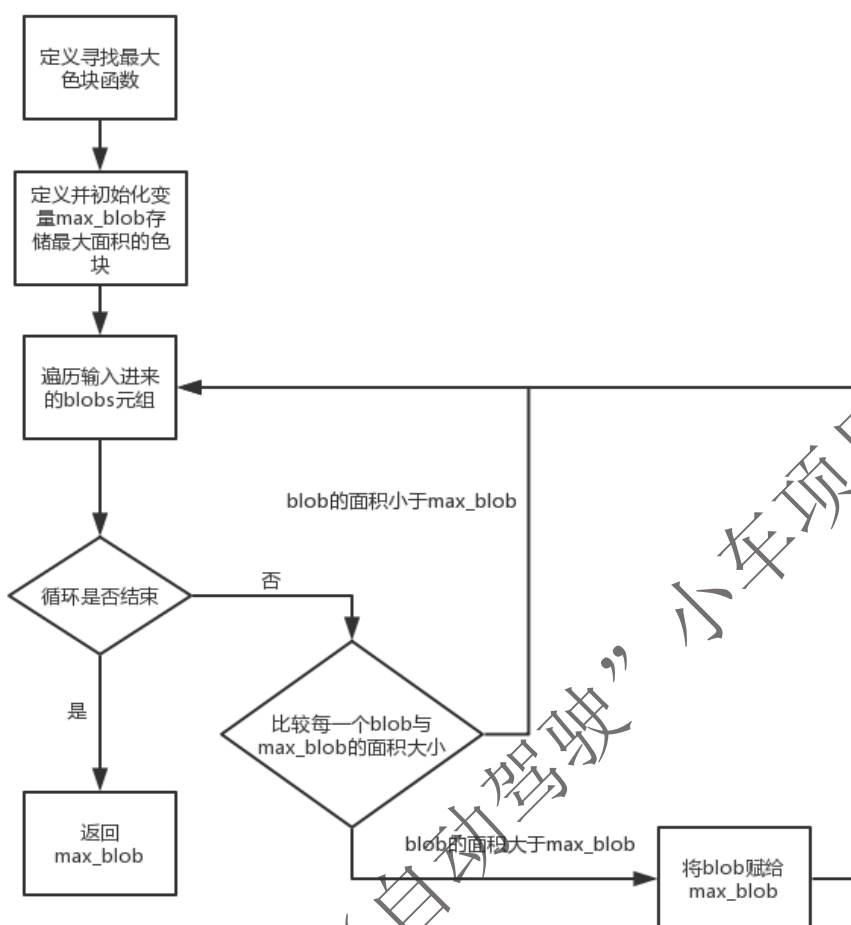
在画框时，加上 `merge=True` 语句，那么就会有多个 `blob` 被合并到一个 `blob`

参考代码如下：

```
Image.draw_rectangle(blob[0:4],merge = True)
```

方法二：（该方法适用于选取整张图片中最大的色块）

思路如下：



---

# 视觉循迹项目书

上一章节我们对 OpenMV 进行了编程，本章节我们将把 OpenMV 和小车联系起来，通过通信把 OpenMV 处理完的数据传递给小车，并让小车对接收到的数据做出“应激性”，赋予小车智能的效果，最终达到视觉巡线的效果。

## 目标

1. 了解并使用串口通信
2. 确定待传递数据的格式和拆包还原数据
3. 实现小车面对不同颜色的车道发出不同的声音，最终实现循迹效果

## 构建

### 目标一：串口通信

串口通信是指外设和计算机间，通过数据信号线、地线、控制线等，按位进行传输数据的一种通讯方式。这种通信方式使用的数据线少，在远距离通信中可以节约通信成本。串口是计算机上一种非常通用的设备通信协议。pyserial 模块封装了 python 对串口的访问，为多平台的使用提供了统一的接口。因此，有了 pyserial 库之后，我们只需进行调用相关函数就可以完成 OpenMV 与主机间的数据交换。

#### 1. 打开端口

---

pyserial 模块内置了很多接口类型，我们可以有很多不同的选择，使用格式如下：

```
ser=serial.Serial(portx, bps, timeout)
```

portx 是连接的端口名称，例如 GNU / Linux 上的 / dev / ttyUSB0 等 或 Windows 上的 COM3 ，在本项目中我们使用的是 Linux 连接端口；

bps 是波特率，表示每秒钟传送的码元符号的个数，是衡量数据传送速率的指标，它用单位时间内载波调制状态改变的次数来表示。在信息传输通道中，携带数据信息的信号单元叫码元，每秒钟通过信道传输的码元数称为码元传输速率，简称波特率。波特率是传输通道频宽的指标， 可以设置的范围 9600 到 115200。

timeout 为超时设置, None: 永远等待操作, 0 为立即返回请求结果，其他值为等待超时时间(单位为秒)

例如，在本项目中可以使用如下代码：

```
ser=serial.Serial("/dev/ttyUSB0",9600,timeout=0.5)
```

## 2.读取数据

使用 read() 函数进行数据读取，例如：

```
s = ser.read(10)#从端口读 10 个字节
```

data = ser.readline().decode('utf-8')#读一行，以 /n 结束，要是没有 /n 就一直读，直到阻塞为止，且以 utf-8 格式编码。

目标二：确定待传递数据的格式和拆包还原数据

---

当使用 OpenMV 发送单个数据时，我们可以选择发送一个数据，那如果我们需要一次发送多个变量呢？例如，在真实的自动驾驶场景中，我们不仅要发送巡线坐标，还要发送红绿灯的信息，那么当我们面临多个数据时，我们就可以使用列表进行多个数据的传输，也就是“打包”操作，并且可以使用列表的索引操作，进行数据的精确提取。

（列表的使用详见 PXX）

在使用 serial 函数进行传输时，函数的返回值是一字符串格式，但是我们需要的是整数类型，因此需要对数据进行“拆包”操作还原数据。

下面简要介绍如何使用 Python 语言来编码和解码 JSON 对象。

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式，易于人阅读和编写。使用 JSON 函数需要导入 json 库：

```
import json
```

两个常用函数

json.dumps() : 将 Python 对象编码成 JSON 字符串

json.loads() : 将已编码的 JSON 字符串解码为 Python 对象

例如，在本项目中：

```
json_data = json.loads(data)#将列表元素转变为整数类型
```

### Tips:

1.在进行通信的过程中，可能会发生丢失数据包的情况，从而导致程序崩溃，为此我们需要使用 try-except 语句进行错误处理，试着

---

使用 `pass` 语句来处理产生 `data` 可能产生的 `ValueError` 错误。

2.缓冲区问题，一般在进行读写操作的时候，数据是先被读到了内存中，再把数据写到文件中，当你数据读完的时候不代表你的数据已经写完了，因为还有一部分有可能会留在内存这个缓冲区中。

`flushInput()` 函数可以把缓冲区的数据强行输出，主要用在 IO 中，即清空缓冲区数据。

目标三：视觉循迹

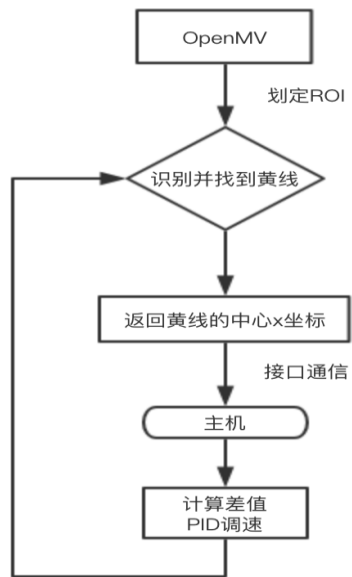
1.为了检验是否能够完成之前的通信目标，我们可以进行一个小测验：

实现对道路引导线的颜色识别，并且使用 `sound` 模块让小车发出声音：例如，在看到黄线时，发出“yellow line”，黑线时发出“black line”的声音。

Tips: 可以使用 `ev3.sound` 类，`speak` 方法进行发声。（详见 PXX）

2.使用之前的 PID 控制算法程序，更改数据获取方式，进行巡线。

实施流程图如下：



上海师范大学——“自动驾驶”小车项目组



---

# 司机上路

本册课本知识学习完成后，我们尝试利用学过的知识完成此综合测评。

## 任务

1. 循直线并将小物块送至加油站
2. 从加油站循曲线回基地
3. 识别到基地挡板停下
4. 发出任务完成提示音
5. 使用摄像头（OpenMV）巡线
6. 在规定时间内走完全程

上海师范大学—“自动驾驶”小车项目组

上海师范大

