

第 1 章

计算机视觉概述

计算机视觉处理是当今信息科学中发展最快的热点研究方向，涉及光学、电子和计算机科学等多个学科。计算机科学中的数字图像处理是其重要基础。本章将阐述数字图像处理的重要基础以及计算机视觉的基本概念。这些理论概念虽然重要，但是如果全面展开来讲，不但内容繁多（至少一本书），而且读者会在有限的时间内由于对抽象理论感到枯燥而逐渐失去学习兴趣。因此，本书将这些理论知识浓缩为一章，为后面章节做一个理论铺垫。在后续学习过程中，一旦看到理论术语，只需翻阅第1章即可，以此方便读者阅读本书。如果是从来没有接触过OpenCV开发的读者，更要学习一下本章内容，从而对图像处理有感性认识。理论的东西，不能一开始就贪多，贪多肯定会感到枯燥，从而放弃；但也不能完全没有，否则就是沙滩筑高楼。

1.1 图像的基本概念

1.1.1 图像和图形

图像是指使用照相机、摄影机、扫描仪等输入设备捕捉实际的画面产生的数字图像，是由像素点阵构成的位图。图像是通过对物体和背景的“摄取”而取得的，这里的“摄取”指的是一种“记录”过程。图像是对客观世界的反映，“图”是指物体透射光或反射光的分布，“像”是人的视觉对“图”的认识。“图像”是两者的结合，它既是一种光的分布，又包含人的视觉心理因素。

图形是用数学规则产生的或具有一定规则的图案，是由外部轮廓线条构成的矢量图。图形往往用一组符号或线条来表示，例如房屋设计图就是用线条来表现房屋结构的。图形用一组指令集合来描述内容，如描述构成该图的各种图元位置、维数、形状等，描述对象可任意缩放而不会失真。图像和图形在一定条件下可相互转换。

1.1.2 数字图像及其特点

数字图像又称数码图像或数位图像，是二维图像用有限数字、数值像素的表示。它由数组或矩阵表示，其中每个元素代表图像的一个像素，表示其光照位置和强度都是离散的。数字图像是由模拟图像数字化得到的，以像素为基本元素的，可以用数字计算机或数字电路存储和处理的图像。

数字图像可以由许多不同的输入设备和技术生成，例如数码相机、扫描仪、坐标测量机等，也可以从任意的非图像数据合成得到，例如数学函数或者三维几何模型（计算机图形学的一个主要分支）。数字图像处理领域主要研究数字图像的变换算法。

数字图像有如下几个特点：

1) 信息量大

以像素数目较少的电视图像为例，它一般是由 512×512 个像素（8bit）组成，其总数据量为 $512 \times 512 \times 8\text{bit} = 2097152\text{bit} = 262144\text{B} = 256\text{KB}$ 。这么大的数据量必须由计算机处理，并且计算机内存容量要大。为了运算方便，常需要几倍其数据量的内存。

2) 占用频带宽大

一般语言信息（如电话、传真、电传、电报等）的带宽仅4kHz左右，而图像信息所占 frequencies 的带宽要大3个数量级。例如，普通电视的标准带宽是6.5MHz，等于语言带宽的14倍。在摄像、传输、存储、处理、显示等各环节的实现上技术难度大，因而对频带的压缩技术的要求是很迫切的。

3) 相关性大

每幅图像中相邻像素之间不独立，并且具有很大的相关性，有时大片大片的像素间具有相同或接近的灰度。例如，就电视画面而言，前后两幅图像的相关系数往往在0.95以上。因此，压缩图像信息的潜力很大。

4) 非客观性

图像信息的最终接收器是人的视觉系统。由于图像信息和视觉系统十分复杂，与环境条件、视觉特性、情结、精神状态、知识水平等有关，这就要求图像系统与视觉系统具有良好的“匹配”，因此必须研究图像的统计规律和视觉特征。

1.1.3 图像单位

任意一幅数字图像粗看起来似乎是连续的，实际上是不连续的，它由许多密集的色点组成。就像任意物质一样，肉眼看上去是连续的，但实质上都是由分子组成的。这些色点是构成一幅图像的基本单元，被称为像素（或像素点、像元，Pixel）。例如，一幅图片由30万个色点组成，那这幅图片的像素就是30万。像素是数字图像的基本元素。显然，像素越多，画面就越清晰。

像素是感光元件记录光信号的基本单位，通常来说1个像素对应1个光电二极管。我们常说相机是多少像素，这个像素就是说这款相机的感光元件有多少个，有100万个感光元件的相

机就是100万像素的相机，有4000万个感光元件的相机就是4000万像素的相机，以此类推。一台100万像素的相机拍摄的照片洗成5寸比洗成6寸清晰一点。

像素是在模拟图像数字化时对连续空间进行离散化得到的。每个像素具有整数行（高）和列（宽）位置坐标，同时每个像素都具有整数灰度值或颜色值。

如图1-1所示的图片尺寸可以从其文件属性窗口（见图1-2）中查到，是500×338，即图片是由一个500×338的像素点矩阵构成的，这幅图片的宽度是500像素，高度是338像素，共有169000（500×338）个像素点。



图 1-1

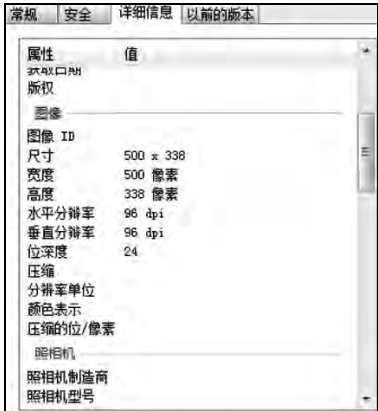


图 1-2

又如，屏幕分辨率是1024×768，也就是说设备屏幕的水平方向上有1024个像素点，垂直方向上有768个像素点。像素的大小是没有固定长度的，不同设备上一个单位像素色块的大小是不一样的。例如，尺寸面积大小相同的两块屏幕，分辨率大小可以是不一样的，分辨率高的屏幕上像素点（色块）就多，所以屏幕内可以展示的画面就更细致，单个色块面积更小；分辨率低的屏幕上像素点（色块）更少，单个像素面积更大，显示的画面就没那么细致。

1.1.4 图像分辨率与屏幕分辨率

图像分辨率是指每英寸图像内的像素点数，单位是像素每英寸。分辨率越高，像素的点密度越高，图像就越逼真（做大幅喷绘时，要求图片分辨率要高，就是为了保证每英寸的画面拥有更多的像素点）。

屏幕分辨率是屏幕每行的像素点数乘以每列的像素点数。每个屏幕都有自己的分辨率，屏幕分辨率越高，所呈现的色彩就越多，清晰度也越高。

1.1.5 图像的灰度与灰度级

把白色与黑色之间按对数关系分为若干等级，称为灰度。灰度分为256阶，0为黑色。灰度就是没有色彩，RGB色彩分量全部相等，如RGB(100,100,100)代表灰度为100，RGB(50,50,50)代表灰度为50。

一幅图像中不同位置的亮度是不一样的，可用 $f(x,y)$ 来表示点 (x,y) 上的亮度。由于光是一种能量形式，因此亮度是非负有限的 ($0 \leq f(x,y) < \infty$)。在图像处理中，常用灰度和灰度

级这个名称。在单色图像中，坐标 (x,y) 点的亮度称为该点的灰度或灰度级。设灰度为 L ，则 $L_{\min} \leq L \leq L_{\max}$ 。区间 $[L_{\min}, L_{\max}]$ 称为灰度范围。

在室内处理图像时，一般 $L_{\min} \approx 0.005\text{Lux}$ ， $L_{\max} \approx 100\text{Lux}$ ，其中Lux是勒克斯（照明单位）。实际使用中，把这个区间规格化为 $[0, L_{\max}]$ 。其中， $L_{\min}(0)$ 为黑色， $L_{\max}$ 为白色，所有在白色和黑色之间的值代表连续变化的灰度。

灰度级表明图像中不同灰度值的最大数量。灰度级越大，图像的亮度范围越大。灰度级有时会与灰度混淆。灰度（值）表示灰度图像单个像素点的亮度值，值越大，像素点越亮，反之越暗。



图 1-3

灰度级表示灰度图像的亮度层次，比如第1级、第2级……第255级，如图1-3所示。

在图1-3中，第0级的灰度是0，第1级的灰度是8，第2级的灰度是16……每个等级都对应着某个灰度值。级数越多，图像的亮度范围越大，层次越丰富。有时，把最大级数称为一幅图像的灰度级数。

1.1.6 图像的深度

图像深度是指存储每个像素所用的位数，也用于量度图像的色彩分辨率。图像深度确定彩色图像的每个像素可能有的颜色数，或者确定灰度图像的每个像素可能有的灰度级数。它决定了彩色图像中可出现的最多颜色数，或灰度图像中的最大灰度等级。比如，一幅单色图像，若每个像素有8位，则最大灰度数目为2的8次方，即256。一幅彩色图像RGB三个分量的像素位数分别为4、4、2，则最大颜色数目为2的 $(4+4+2)$ 次方，即1024，也就是说像素的深度为10位，每个像素可以是1024种颜色中的一种。

例如，一幅画的尺寸是 1024×768 像素，深度为16，则它的数据量为1.5MB，计算如下：

$$\begin{aligned} 1024 \times 768 \times 16\text{bit} &= (1024 \times 768 \times 16) \div 8\text{B} \\ &= [(1024 \times 768 \times 16) / 8] \div 1024\text{KB} \\ &= \{[(1024 \times 768 \times 16) \div 8] \div 1024\} \div 1024\text{MB} \end{aligned}$$

1.1.7 二值图像、灰度图像与彩色图像

二值图像（Binary Image）上的每一个像素只有两种可能的取值或灰度等级状态，可用黑白、B&W、单色图像表示。按名字来理解，二值图像只有两个值，即0和1，0代表黑，1代表白，或者说0表示背景，1表示前景。其保存也相对简单，每个像素只需要1bit就可以完整存储信息。如果把每个像素看成随机变量，一共有 N 个像素，那么二值图像有2的 N 次方种变化，而8位灰度图有255的 N 次方种变化，8位三通道RGB图像有 $255 \times 255 \times 255$ 的 N 次方种变化。也就是说同样尺寸的图像，二值图像保存的信息更少。

灰度图像（Gray Scale Image）又称灰阶图像，是指用灰度表示的图像。除了常见的卫星图像、航空照片外，许多地球物理观测数据也以灰度图表示。我们平时看到的灰度图像是由0到255个像素组成的。

灰度图像是二值图像的进化版本，是彩色图像的退化版，也就是灰度图保存的信息没有彩色图像多，但比二值图像多。灰度图像只包含一个通道的信息，而彩色图像通常包含三个通道的信息，单一通道可以理解为单一波长的电磁波，所以红外遥感、X断层成像等单一通道电磁波产生的图像都为灰度图。在实际应用中，灰度图具有易于采集和传输等特性，因此基于灰度图像开发的算法非常丰富。

灰度图像的每个像素只有一个采样颜色，这类图像通常显示为从最暗黑色到最亮白色的灰度，理论上这个采样可以是任何颜色的不同深浅，甚至可以是不同亮度上的不同颜色。灰度图像与黑白图像不同，在计算机图像领域中黑白图像只有黑色与白色两种颜色，但是灰度图像在黑色与白色之间还有许多级的颜色深度。灰度图像经常是在单个电磁波频谱（如可见光）内测量每个像素的亮度得到的，用于显示的灰度图像通常用每个采样像素8位的非线性尺度来保存，这样可以有256级灰度（如果用16位，则有65536级）。

彩色图像也就是RGB图像，每个像素由3个通道进行表示。彩色图像的每个像素通常是由红（R）、绿（G）、蓝（B）3个分量来表示的，分量值范围是(0,255)。

1.1.8 通道

通道表示把图像分解成一个或多个颜色成分，通常可以分为单通道、三通道和四通道。

- 单通道：一个像素点只需要一个数值表示。单通道只能表示灰度，0 为黑色。单通道图像就是图像中每个像素点只需一个数值表示。
- 三通道：把图像分为红、绿、蓝 3 个通道。三通道可以表示彩色，其中全 0 表示黑色。
- 四通道：在 RGB 基础上加上 alpha 通道。alpha 通道表示透明度，值为 0 时表示全透明。

1.1.9 图像存储

在计算机中用 $M \times N$ 的矩阵表示一幅尺寸大小为 $M \times N$ 的数字图像，矩阵元素的值就是该图像对应位置上的像素值。三通道图像数据在内存中的存储是连续的，每个通道元素按照矩阵行列顺序进行排列。通常计算机按照RGB方式存储三通道图像格式，而图像采集设备输出图像格式一般是BGR方式。

1.2 图 像 噪 声

1.2.1 图像噪声的定义

图像噪声可以理解为妨碍人的视觉器官或系统传感器对所接收的图像源信息进行理解或分析的各种因素。一般图像噪声是不可预测的随机信号，只能用概率统计的方法去认识。噪声作用于图像处理的输入、采集、处理以及输出的全过程，特别是图像在输入、采集的过程中引入的噪声，会影响图像处理的全过程，以至于影响输出结果。噪声对图像的影响无法避免，因此一个良好的图像处理系统，无论是模拟处理还是计算机处理，无一不将最前一级

的噪声减小到最低作为主攻目标。因此，滤除图像中的噪声就成为图像处理中极为重要的步骤，对图像处理有着重要的意义。

数字图像的噪声主要来源于图像获取的数字化过程。图像传感器的工作状态受各种因素的影响，如环境条件、传感器元件质量等。在图像传输的过程中，所用的传输信道受到干扰，也会产生噪声污染。例如，通过无线网络传输的图像可能会因为光或其他大气因素的干扰而受到噪声污染。图像噪声的种类有多种，包括高斯噪声、瑞利噪声、伽马噪声、指数噪声、均匀噪声以及脉冲噪声（又称为椒盐噪声或双极性噪声）等。其中，脉冲噪声在图像噪声中最为常见。在图像生成和传输的过程中，经常会产生脉冲噪声，主要表现在成像的短暂停留中。脉冲噪声对图像质量有较大的影响，需要采用图像滤波方法给予滤除。

1.2.2 图像噪声的来源

外部噪声是指系统外部干扰以电磁波的方式或经电源串进系统内部而引起的噪声，例如电气设备、天体放电现象等引起的噪声。

内部噪声一般可分为以下4种：

（1）由光和电的基本性质引起的噪声。例如，电流是由电子或空穴粒子的集合定向运动形成的，而这些粒子运动的随机性会形成散粒噪声；导体中自由电子的无规则热运动会形成热噪声；根据光的粒子性，图像是由光量子传输的，而光量子密度随时间和空间变化会形成光量子噪声等。

（2）电器的机械运动产生的噪声。例如，各种接头因抖动而引起电流变化所产生的噪声，如磁头、磁带等抖动或一起的抖动等。

（3）器材材料本身引起的噪声。例如，正片和负片的表面颗粒性以及磁带、磁盘表面缺陷所产生的噪声。随着材料科学的发展，这些噪声在不断减少，但目前来讲，还是不可避免。

（4）系统内部设备电路所引起的噪声。例如，电源引入的交流噪声，偏转系统和箝位电路所引起的噪声等。

1.2.3 图像噪声的滤除

通过平滑图像，可以有效减少和消除图像中的噪声，从而改善图像质量，这对于提取对象特征以进行分析非常有利。经典的平滑技术通常使用局部算子对噪声图像进行处理。在对某个像素进行平滑处理时，仅对其局部小邻域内的其他像素进行操作。这种方法的优点在于计算效率高，并且能够实现多个像素的并行处理。近年来，出现了一些新的图像平滑处理技术，这些技术结合了人眼的视觉特性，运用了模糊数学理论、小波分析、数学形态学和粗糙集理论等新方法，取得了良好的效果。

灰度图像常用的滤波方法主要分为线性和非线性两大类。线性滤波方法一般通过取模板做离散卷积来实现。这种方法在平滑脉冲噪声点的同时，也会导致图像模糊，从而损失图像细节信息。非线性滤波方法中应用最多的是中值滤波，中值滤波可以有效地滤除脉冲噪声，具有相对好的边缘保持特性，并易于实现，因此被公认为一种有效的方法。然而，中值滤波同时也

会改变未受噪声污染的像素的灰度值，使图像变得模糊。随着滤波窗口长度的增加和噪声污染的加重，中值滤波效果明显下降。

针对中值滤波方法的缺陷，目前科学家已经提出了一些改进方法，但这些方法都是无条件地对所有的输入样本进行滤波处理。然而，对于一幅噪声图像来说，只有一部分像素受到了噪声的干扰，其余的像素仍保持原值。无条件地对每个像素进行滤波处理必然会损失图像的某些原始信息。因此，人们提出在滤波处理中加入判断的过程，即首先检测图像的每个像素是否为噪声，然后根据噪声检测结果进行切换，输出结果在原像素灰度和中值滤波或其他滤波器计算结果之间切换。由于是有选择地进行滤波处理，避免了不必要的滤波操作和图像的模糊，因此滤波效果得到了进一步的提高。但这些方法在判断和滤除脉冲噪声的过程中还存在一定的缺陷，比如对于较亮或较暗的图像会产生较多的噪声误判和漏判，甚至无法进行噪声检测，同时算法的计算量也明显增加，从而影响了滤波效果和速度。

既然图像有时不可避免地会产生噪声，那就需要对图像进行处理。

1.3 图像处理

信息是自然界物质运动的一个重要方面，人们认识和改造世界需要各种信息图像。这些信息是人类获取外界知识的主要来源，其中约80%的信息通过人眼获得。在现代科学研究、生产活动等各个领域，越来越多地使用图像信息来认识和判断事物，以解决实际问题。获取图像信息固然重要，但我们的主要目的是对这些信息进行处理，以便从大量复杂的图像数据中提取出感兴趣的信息。图像处理是指对图像信息进行加工和处理，以满足视觉或应用上的需求。因此，从某种意义上来说，对图像信息的处理比图像本身更为重要。

21世纪是一个充满信息的时代，图像作为人类感知世界的视觉基础，是获取、表达和传递信息的重要手段。计算机时代所说的图像处理通常指的是数字图像处理，即利用计算机对图像进行处理。这一技术的发展历史并不悠久，数字图像处理技术的起源可以追溯到20世纪20年代，当时通过海底电缆对从英国伦敦传输到美国纽约的一幅照片采用了数字压缩技术。

数字图像处理技术首先帮助人们以更客观、准确的方式认识世界。人的视觉系统能够帮助人类获取超过3/4的信息，而图像和图形则是所有视觉信息的载体。尽管人眼的辨识能力很高，可以识别上千种颜色，但在很多情况下，图像对于人眼而言是模糊的，甚至是不可见的。通过图像增强技术，可以使这些模糊或不可见的图像变得清晰明亮。

在计算机视觉这一领域诞生的初期，一种普遍的研究范式是将图像看作二维的数字信号，然后借用数字信号处理中的方法进行处理，这就是数字图像处理（Digital Image Processing）。

1.3.1 图像处理的分类

图像处理通常可以分为3类：光学模拟处理、电学模拟处理和计算机数字处理。

1) 光学模拟处理

光学模拟处理也称光信息处理，建立在傅里叶光学基础上，通过光学滤波、相关运算、频谱分析等，可以实现图像像质的改善、图像识别、图像的几何畸变和光度的校正、光信息的编码和存储、图像的伪彩色化、三维图像显示、对非光学信号进行光学处理等。

2) 电学模拟处理

电学模拟处理把光强度信号转换成电信号，然后用电子学的方法对信号进行加、减、乘、除、浓度分割、反差放大、彩色合成、光谱对比等，在电视视频信号处理中经常应用。随着该项技术的日趋成熟和逐步改进，根据电学模拟方法的基本特征和规律，可以将其细分为以下几种功能：

- (1) 通过建立反变化将信息数据进行重组，组成新的排列形式；
- (2) 改变时钟脉冲的变化规律，并通过模拟的方式实现。
- (3) 将各种响应不同的处理模式看作过滤器，完成信号的处理。

电学模拟处理方法具有较低的运行设备和成本投入，具有明显的优势，能够使计算机图像处理技术在较短的时间内完成图像的过滤处理。这种方法的应用前景广阔，具有很高的实用价值。

3) 计算机数字处理

图像的计算机数字处理是在以计算机为中心的、包括各种输入/输出及显示设备在内的数字图像处理系统上进行的。它将连续的模拟图像变换成离散的数字图像后，使用由特定的物理模型和数学模型编制而成的程序进行控制，并实现各种要求的处理。。

1.3.2 数字图像处理

数字图像处理技术，通俗地讲是指应用计算机以及数字设备对图像进行加工处理的技术。通常包括如下几个过程。

1) 图像信息的获取

为了在计算机上进行图像处理，必须把作为处理对象的模拟图像转换成数字图像信息。图像信息的获取一般包括图像的摄取、转换及数字化等几个步骤。这部分主要由处理系统硬件实现。

一般情况下，由于图像处理的设备比较大，不易在室外使用，因此通常输入图像分两步进行：首先在室外通过摄像机、照相机、数码相机等设备将图像记录下来，然后在室内利用输入设备进行输入。一般用磁带记录的是视频信号，通过AN口、1394口输入视频采集卡；用胶片记录的是照片，可通过扫描仪扫描输入；电子照片可直接通过串口、并口或USB口输入。

2) 图像信息的存储与交换

由于数字图像中的信息量庞大，在处理过程中必须对数据进行存储和交换。为了有效解决大数据量与交换和传输时间之间的矛盾，通常除了采用大容量内存存储器进行并行传输和直接存储访问外，还需要利用外部磁盘、光盘和磁带等存储方式，以提高处理效率。这部分主要功能通常由硬件完成。

3) 具体的图像处理

数字图像处理是指将空间上离散的、在幅度上量化分层的数字图像经过特定的数学模式进行加工处理,以获得人眼视觉或某种接收系统所需的图像。自20世纪80年代以来,计算机技术和超大规模集成电路技术的迅猛发展,极大地推动了通信技术(包括语言数据、图像)的飞速发展。因为图像通信具有形象直观、可靠、高效率等一系列优点,尤其是数字图像通信比模拟图像通信更具抗干扰性,便于进行压缩编码处理且易于加密,因此数字处理技术在图像通信工程中获得了广泛应用。

4) 图像的输出和显示

数字图像处理的最终目的是提供便于人眼或接收系统解释和识别的图像,因此图像的输出和显示十分重要。一般图像输出的方式可分为硬拷贝(如照相、打印、扫描等)和软拷贝(如CRT监视器及各种新型的平板监视器等)。

1.3.3 数字图像处理常用方法

数字图像处理常用的方法有图像变换、图像增强、图像分割、图像描述、图像分类(识别)和图像重建等。

1. 图像变换

由于图像阵列很大,直接在空间域中进行处理涉及的计算量很大,因此往往采用各种图像变换的方法,如傅里叶变换、沃尔什变换、离散余弦变换等间接处理技术,将空间域的处理转换为变换域的处理。这样不仅可以减少计算量,而且能获得更有效的处理(如傅里叶变换可在频域中进行数字滤波处理)。目前新兴研究的小波变换在时域和频域中都具有良好的局部化特性,在图像处理中也有着广泛且有效的应用。

图像编码压缩技术可减少描述图像的数据量(比特数),以便节省图像传输和处理的时间,以及减少所占用的存储器容量。压缩可以在不失真的前提下获得,也可以在允许失真的条件下进行。编码是压缩技术中重要的方法,它在图像处理技术中是发展最早且比较成熟的技术。

2. 图像增强

对于一个数字图像处理系统来说,一般可以将处理流程分为3个阶段:首先是图像预处理阶段,其次是特征抽取阶段,最后是识别分析阶段。图像预处理阶段尤为重要,如果这个阶段处理不好,会直接导致后面的工作无法展开。图像增强是图像预处理阶段的重要步骤。

在采集图像时,由于光照的稳定性与均匀性等噪声的影响,灰尘对CCD摄像机镜头的影响,以及图像传输过程中由于硬件设备而获得的噪声,使得获取的图像不够理想,往往存在噪声、对比度不够、目标不清晰、有其他物体干扰等缺点。这就需要用图像增强技术来改善图像效果。

图像增强就是增强图像中用户感兴趣的信息,其主要目的有两个:一是改善图像的视觉效果,提高图像成分的清晰度;二是使图像变得更有利于计算机处理。

图像增强不是以图像保真原则为基点来处理图像的,而是根据图像质量变坏的一般情况提出一些改善方法。例如,在图像处理中,可以采用图像均衡的方法来缩小图像灰度差别,采用平滑滤波的方法去除图像存在的噪声,采用边缘增强的方法改善图像轮廓的不明显。

图像增强主要应用在图像特别暗时，或者因为曝光太亮而无法让目标突出时，这个时候就需要把目标的亮度提高一点，然后把不必要的障碍（俗称噪声）调暗，以利于目标清晰度最大化。

图像增强的方法通过一定手段对原图像附加一些信息或变换数据，有选择地突出图像中感兴趣的特征或者抑制（掩盖）图像中某些不需要的特征，使图像与视觉响应特性相匹配。

在图像增强过程中，不分析图像降质的原因，处理后的图像不一定逼近原始图像。

通过各种手段来获得清晰图像的方法就是图像增强。根据增强的信息不同，图像增强可以分为边缘增强、灰度增强、色彩饱和度增强等。其中，灰度增强又可以根据增强处理过程所在的空间不同，分为空间域增强和频率域增强两大类，分别简称空域法和频域法。

1) 空域法

空域法主要是直接在空间域内对图像进行运算处理，分为点运算算法和邻域去噪算法（也称邻域增强算法）。

点运算通常包括灰度变换和直方图修正等，目的或使图像成像均匀，或扩大图像动态范围，扩展对比度。

邻域去噪算法分为图像平滑和锐化两种。平滑一般用于消除图像噪声，但是也容易引起图像边缘的模糊，常用算法有均值滤波、中值滤波。锐化的目的在于突出物体的边缘轮廓，便于目标识别，常用算法有梯度法、算子、高通滤波、掩码匹配法、统计差值法等。

2) 频域法

频域法是利用图像变换方法将原来的图像空间中的图像以某种形式转换到其他空间中，然后利用该空间的特有性质进行图像处理，最后转换回原来的图像空间中，从而得到处理后的图像。

频域法增强技术的基础是卷积理论。其中，频域变换可以是傅里叶变换、小波变换、DCT变换、Walsh变换等。

我们可以用一幅图来表示图像增强所用的具体方法分类，如图1-4所示。

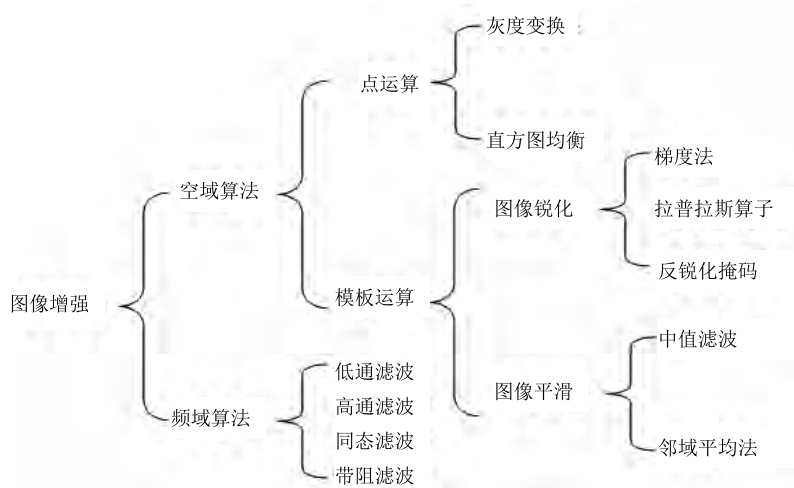


图 1-4

作为初学者，我们也不需要面面俱到，可以先选择重点的几项掌握。

3. 图像分割

图像分割是数字图像处理中的关键技术之一，它将图像中有意义的特征部分提取出来。有意义的特征有图像中的边缘、区域等，这是进一步进行图像识别、分析和理解的基础。虽然目前已研究出不少边缘提取、区域分割的方法，但还没有一种普遍适用于各种图像的有效方法。因此，对图像分割的研究还在不断深入之中，是目前图像处理研究的热点之一。

4. 图像描述

图像描述是图像识别和理解的必要前提。作为简单的二值图像，可采用其几何特性描述物体的特性。一般图像的描述方法采用二维形状描述，可分为有边界描述和区域描述两类。对于特殊的纹理图像，可采用二维形状描述。随着图像处理研究的深入发展，已经开始进行三维物体描述的研究，提出了体积描述、表面描述、广义圆柱体描述等方法。

5. 图像分类（识别）

图像分类（识别）属于模式识别的范畴，其主要内容是对经过某些预处理（增强、复原、压缩）的图像进行分割和特征提取，从而进行判决分类。图像分类常采用经典的模式识别方法，有统计模式分类和句法（结构）模式分类。近年来新发展起来的模糊模式识别和人工神经网络模式分类在图像识别中越来越受重视。

6. 图像重建

图像重建是指对一些三维物体，应用X射线、超声波等物理方法取得物体内部结构数据，再将这些数据进行运算处理而构成物体内部某些部位的图像。目前图像重建成功的例子是CT技术（计算机断层扫描成像技术）、彩色超声波等。这是图像处理的另一个发展方向。

1.3.4 图像处理的应用

图像处理的应用十分广泛，大大促进了现代社会的发展，比如人脸支付、指纹支付就用到图像处理，停车场识别车牌也用到图像处理。下面用表1-1来简要说明图像处理的常见应用。

表 1-1 图像处理的常见应用

领 域	应用内容
物理化学	结晶分析、谱分析
生物医学	细胞分析、染色体分类、血球分类、X 光、CT
环境保护	水质及大气污染调查
地质	资源勘探、地图绘制
农林	植被分布调查、农作物估产
海洋	鱼群探查
水利	河流分布、水利及水害调查
气象	云图分析、灾害性检测等
通信	传真、电视、可视电话图像通信

(续表)

领 域	应用内容
工业	工业探伤、计算机视觉、自动控制、机器人
法律	公安指纹识别、人像鉴定
交通	铁路选定、交通指挥、汽车识别
军事	侦察、成像融合、成像制导
宇航	星际探险照片处理
文化	多媒体、动画特技

1.4 计算机视觉概述

1.4.1 基本概念

计算机视觉是一门研究如何使机器“看”的科学，更进一步地说就是指用摄像机和计算机代替人眼对目标进行识别、跟踪和测量等机器视觉，并进一步进行图形处理，使用计算机处理成为更适合人眼观察或传送给仪器检测的图像。作为一个科学学科，计算机视觉研究相关的理论和技术，试图建立能够从图像或者多维数据中获取“信息”的人工智能系统。这里的信息指的是由Shannon定义的，可以用来帮助做“决定”的信息。因为感知可以看作从感官信号中提取信息，所以计算机视觉也可以看作研究如何使人工系统从图像或多维数据中“感知”的科学。

计算机视觉就是用各种成像系统代替视觉器官作为输入敏感手段，并由计算机来代替大脑完成处理和解释。计算机视觉的最终研究目标是使计算机能像人那样通过视觉观察和理解世界，具有自主适应环境的能力。这是要经过长期的努力才能达到目标，因此，在实现最终目标以前，人们努力的中期目标是建立一种视觉系统，这个系统能依据视觉敏感和反馈的某种程度的智能来完成一定的任务。例如，计算机视觉的一个重要应用领域就是自主车辆的视觉导航，还没有条件实现像人那样能识别和理解任何环境、完成自主导航的系统。因此，人们努力的研究目标是实现在高速公路上具有道路跟踪能力，可避免与前方车辆碰撞的视觉辅助驾驶系统。这里要指出的一点是，在计算机视觉系统中，计算机起代替人脑的作用，但这并不意味着计算机必须按人类视觉的方法完成视觉信息的处理。计算机视觉可以根据计算机系统的特点来进行视觉信息的处理。但是，人类视觉系统是迄今为止人们所知道的功能最强大和完善的视觉系统。计算机视觉通过各种成像系统代替视觉器官作为输入敏感手段，并由计算机来代替大脑完成处理和解释。计算机视觉的最终研究目标是使计算机能像人那样通过视觉观察和理解世界，具有自主适应环境的能力。这是需要经过长期的努力才能达到目标，因此，在实现最终目标以前，人们努力的中期目标是建立一种视觉系统，这个系统能依据视觉敏感和反馈的某种程度的智能来完成一定的任务。例如，计算机视觉的一个重要应用领域是自主车辆的视觉导航，这还需要长期努力才能达到目标。因此，人们正在努力实现具有道路跟踪能力的视觉辅助驾驶系统，以避免与前方车辆的碰撞。值得注意的是，虽然计算机在计算机视觉系统中起代替人脑的作用，但这并不意味着计算机必须按照人类视觉的方法来完成视觉信息的处理。计算机视觉可以根据计算机系统的特点进行有效的视觉信息处理。人类视觉系统是目前已知的功能最强大和完善的

视觉系统，它在许多方面都具有重要的优势。但为了适应特定的环境和任务，我们还需要对计算机视觉技术进行进一步的优化和改进。

计算机视觉是一门综合性的学科，吸引了来自各个学科的研究者参与到对它的研究之中，其中包括计算机科学和工程、信号处理、物理学、应用数学和统计学、神经生理学和认知科学等。

计算机视觉包括图像处理和模式识别，除此之外，还包括空间形状的描述、几何建模以及认识过程（认知科学与神经科学）。实现图像理解是计算机视觉的终极目标。

- 图像处理：图像处理技术把输入图像转换成具有所希望特性的另一幅图像。例如，可通过处理使输出图像具有较高的信噪比，或通过增强处理突出图像的细节，以便于操作员检验。在计算机视觉研究中经常利用图像处理技术进行预处理和特征抽取。
- 模式识别：模式识别技术根据从图像抽取的统计特性或结构信息对图像进行分类，例如文字识别或指纹识别。在计算机视觉中，模式识别技术经常用于对图像中的某些部分（如分割区域）进行识别和分类。
- 图像理解：对于给定的一幅图像，图像理解程序不仅要描述图像本身，还要描述和解释图像所代表的景物，以便对图像代表的内容做出决定。在人工智能视觉研究的初期，经常使用景物分析这个术语，以强调二维图像与三维景物之间的区别。图像理解除了需要复杂的图像处理以外，还需要具有关于景物成像的物理规律的知识以及与景物内容有关的知识。

认知科学与神经科学（Cognitive Science and Neuroscience）将人类视觉作为主要的研究对象。计算机视觉中已有的许多方法与人类视觉极为相似。许多计算机视觉研究者对研究人类视觉计算模型比研究计算机视觉系统更感兴趣，他们希望计算机视觉更加自然化，更加接近生物视觉。

计算机视觉的研究与人类视觉的研究密切相关。为实现建立与人的视觉系统相类似的通用计算机视觉系统的目标，需要建立人类视觉的计算机理论。

1.4.2 计算机视觉的应用

人类正在进入信息时代，计算机也正越来越广泛地进入各个领域。一方面是更多未经计算机专业训练的人也需要应用计算机，另一方面是计算机的功能越来越强，使用方法越来越复杂。人可以通过视觉、听觉和语言与外界交换信息，并且可用不同的方式表示相同的含义，而计算机却要求严格按照各种程序语言来编写程序，只有这样计算机才能运行。这就使人与人交谈和通信时的灵活性与在使用计算机时所要求的严格和死板之间产生了尖锐的矛盾。为使更多的人能使用复杂的计算机，必须改变过去那种让人来适应计算机、死记硬背计算机使用规则的情况，而是反过来让计算机来适应人的习惯和要求，以人所习惯的方式与人进行信息交换，也就是让计算机具有视觉、听觉和说话等能力。这时计算机必须具有逻辑推理和决策的能力。具有上述能力的计算机就是智能计算机。

智能计算机不但使计算机更便于为人们所使用，而且如果用这样的计算机来控制各种自动化装置（特别是智能机器人），就可以使这些自动化系统和智能机器人具有适应环境和自主做出决策的能力。这就可以在各种场合取代人的繁重工作，或代替人到各种危险和恶劣环境中完成任务。

计算机视觉和机器视觉领域有显著的重叠。计算机视觉涉及被用于许多领域的自动化图像分析的核心技术。机器视觉是人工智能的一个分支。简单来说，机器视觉就是用机器代替人眼来做测量和判断。在许多计算机视觉应用中，计算机被预编程，以解决特定的任务。计算机视觉应用的实例包括：

- (1) 控制过程，比如一个工业机器人。
- (2) 导航，比如自主汽车或移动机器人。
- (3) 检测事件，比如监控视频和人数统计。
- (4) 组织信息，比如图像和图像序列的索引数据库。
- (5) 造型对象或环境，比如医学图像分析系统或地形模型。
- (6) 相互作用，比如输入一个装置时用于计算机与人的交互。
- (7) 自动检测，比如在制造业的应用程序。

其中，最突出的应用领域是医疗计算机视觉和医学图像处理。这个领域的特征信息从图像数据中提取，用于患者医疗诊断的目的。通常，图像数据是形式显微镜图像、X射线图像、血管造影图像、超声图像和断层图像中的信息。比如，可以从这样的图像数据中提取肿瘤、动脉粥样硬化或其他恶性变化。它也可以是器官的尺寸、血流量等。这种应用领域还支持提供新的信息、医学研究的测量。计算机视觉在医疗领域的应用还包括增强超声图像或X射线图像，以降低噪声对图像的影响。

计算机视觉第二个应用领域是工业，这个领域中提取的信息用于制造过程。比如用于质量控制，自动检测最终产品的缺陷。

计算机视觉在军事领域的应用也非常广泛，最明显的例子就是探测敌方士兵、车辆和导弹制导。更先进的系统能够根据为导弹指导发送区域，而不是一个特定的目标，并在导弹到达基于本地获取的图像数据的区域目标时做出选择。现代军事概念，如“战场感知”，意味着各种传感器，包括图像传感器，提供了丰富的作战场景，可用于支持战略决策。在这种情况下，数据的自动处理可以减少复杂性和融合来自多个传感器的信息，以提高可靠性。

一个较新的应用领域是无人驾驶汽车和无人机。无人驾驶汽车或无人机通常使用计算机视觉进行导航，即它们知道自己在哪里，要去哪里，并能检测障碍物。

1.4.3 与相关学科的区别

计算机视觉、图像处理与图像分析、机器视觉是彼此紧密关联的学科，它们的基础理论大致相同。然而各研究机构、学术期刊、会议及公司往往会把自己归为其中某一个领域，于是各种各样的用来区分这些学科的特征便被提了出来。下面给出其中的一种区分方法。

计算机视觉的研究对象主要是映射到单幅或多幅图像上的三维场景，例如三维场景的重建。计算机视觉的研究很大程度上针对图像的内容。

图像处理与图像分析的研究对象主要是二维图像，可以实现图像的转换，尤其是针对像素级的操作，例如提高图像对比度、边缘提取、去噪声和几何变换（如图像旋转）。这一特征表明，无论是图像处理还是图像分析，研究内容都和图像的具体内容无关。

机器视觉主要是指工业领域的视觉研究，例如自主机器人的视觉，用于检测和测量的视觉。

模式识别使用各种方法从信号中提取信息，主要运用统计学的理论。此领域的一个主要方向便是从图像数据中提取信息。

1.5 OpenCV概述

OpenCV (Open Source Computer Vision Library, 开源计算机视觉库) 是一个基于BSD许可(开源)发布的跨平台计算机视觉和机器学习软件库, 可以运行在Linux、Windows、Android和Mac OS操作系统上。它由一系列C函数和少量C++类构成, 同时提供了Python、Ruby、MATLAB等语言的接口, 实现了图像处理和计算机视觉方面的很多通用算法, 属于轻量级的、非常高效的软件库。

OpenCV主要倾向于实时视觉应用, 并在可用时利用MMX和SSE指令, 如今也提供对于C#、Ch、Ruby、Go的支持。

OpenCV提供的视觉处理算法非常丰富, 并且部分以C语言编写, 加上其开源的特性, 若处理得当, 则不需要添加新的外部支持, 也可以完整地编译链接生成执行程序, 所以很多人用它来做算法的移植。OpenCV的代码经过适当改写可以正常运行在DSP系统和ARM嵌入式系统中。

OpenCV是一个跨平台的计算机视觉库, 目标是实现实时计算机视觉, 也就是用摄像机和计算机代替人眼对目标进行识别、跟踪以及测量等, 并进一步做图像处理。图像处理又称为影像处理, 是用计算机对图像进行分析, 以达到所需结果的技术, 一般包括图像压缩、增强和复原, 以及匹配、描述和识别。目前所说的图像处理一般是指数字图像处理。计算机视觉与图像处理的区别主要在于: 计算机视觉的侧重点在于使用计算机来模拟人的视觉, 对客观事物进行“感知”; 图像处理的侧重点在于“处理”, 提取所需要的有效信息。这两者相辅相成, 从而使机器可以在一定程度上模拟人的一些行为, 从而使机器更加人性化、智能化。

开源特性以及强大的社区支持使得OpenCV发展极其迅速。OpenCV 1.0正式版本于2006年发布, 可以运行在Mac OS以及Linux平台上, 但是主要提供C的接口。OpenCV 2.0版本于2009年发布, 代码已显著优化, 同时带来了全新的C++函数的接口, 将其能级无限放大, 使开发者使用更加方便; 另外, 增加了新的平台支持, 包括iOS和Android, 通过CUDA和OpenGL实现了GPU加速; 在编程语言方面, 还为Python和Java用户提供了接口。2014年8月, OpenCV 3.0 Alpha发布, 重大革新之处在于OpenCV 3.0改变了项目架构的方式。之前的OpenCV是一个相对于整体的项目, 各个模块都是以整体的形式构建然后组合在一起; OpenCV 3.0抛弃了整体架构, 使用内核+插件的架构形式, 更加轻量化。OpenCV随着工业4.0与机器人无人机的的发展, 已经在应用领域得到了广泛应用, 有越来越多的从事机器视觉与图像处理的开发者选择OpenCV作为开发工具。

计算机视觉市场巨大而且持续增长, 但这方面没有标准API, 如今的计算机视觉软件大概有以下3种:

- (1) 研究派代码(慢, 不稳定, 独立, 与其他库不兼容)。

(2) 耗费很高的商业化工具（比如Halcon、MATLAB+Simulink）。

(3) 依赖硬件的一些特别的解决方案（比如视频监控、制造控制系统、医疗设备）。

标准的API将简化计算机视觉程序和解决方案的开发，OpenCV致力于成为这样的标准API。

OpenCV致力于真实世界的实时应用，通过优化C代码提升其执行速度，并且可以通过购买Intel的IPP（Integrated Performance Primitives，高性能多媒体函数库）得到更快的处理速度。

OpenCV的应用领域非常广泛，比如人机互动、物体识别、图像分割、人脸识别、动作识别、运动跟踪、机器人、运动分析、机器视觉、结构分析、汽车安全驾驶、军工、卫星导航等。可以说学好了OpenCV，就业和职业发展前景广阔！

目前OpenCV最新版是4.10，它发布于2024年6月，官网相关发布宣传图如图1-5所示。



图 1-5

OpenCV最显著的优化就是从这个4.10版本开始，对JPEG图像的读取和解码有了77%的速度提升，超过了scikit-image、imageio、pillow等工具。其他改进要点如下：

(1) dnn模块的改进，包括：

- 改善内存消耗。
- 增加了将模型转储为与 Netron 工具兼容的 ptxt 格式的功能。
- 支持多个新的 TFlite、ONNX 和 OpenVINO 层。
- 改进了现代 Yolo 探测器支持。
- 添加了 cuDNN 9+和 OpenVINO 2024 支持。

(2) core模块的改进，包括：

- 为 cv::Mat 模块添加了 CV_FP16 数据类型。
- 扩展了 HAL API，用于 minMaxIdx、LUT、meanStdDev 和其他函数。

(3) imgproc模块的改进，包括：

- 为 cv::remap 模块添加了相对位移场选项。
- 重构 findContours 和 EMD。
- 扩展了 HAL API，用于 projectPoints、equalizeHist、Otsu 阈值和其他功能。
- 添加了针对现代 ARMv8 和 ARMv9 平台优化的新底层 HAL 库（KleidiCV）。

- (4) 支持CUDA 12.4+。
- (5) 添加了zlib-ng作为经典zlib的替代品。
- (6) 对Wayland、Apple VisionOS和Windows ARM64的实验性支持。
- (7) OpenCV Model Zoo提供跨平台的预训练深度学习模型。其新增功能包括：
 - 支持更多的模型结构，例如新的卷积架构或者神经网络架构。
 - 提升模型的性能，可能通过模型优化或者使用更高效的实现方式。
 - 提供更多的预处理和后处理的选项，以便用户可以更灵活地使用这些模型。
 - 增加对新硬件或者新框架的支持，例如新版的 TensorRT 或是 ONNX Runtime。

第 2 章

OpenCV的Python开发环境搭建

2.1 Python下载与安装

Python是一种面向对象的解释型计算机程序设计语言，是纯粹的自由软件，遵循GPL（General Public License）协议。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，强调“段落”形式。其第一个公开版发行于1991年。

Python的万能之处在于能够把用其他语言制作的各种模块（尤其是C/C++制作的模块）轻松地连接在一起，因此也常被称为“胶水语言”。常见的一种应用情形是使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分用更合适的语言改写，比如用C/C++重写，而后封装为Python可以调用的扩展类库。

人工智能技术的火热使得Python的支持库越来越丰富、强大。需要注意的是，在使用扩展类库时，可能需要考虑平台问题，某些类库可能不提供跨平台的实现。

在使用OpenCV和Python进行图像处理时，需要注意OpenCV和Python版本之间的兼容性。不同版本的OpenCV可能对应不同的Python版本。因此，在开始使用之前，需要确保OpenCV和Python的版本是兼容的。

以下是常见的OpenCV和Python版本对应关系：

- OpenCV 2.x: 对应 Python 2.7。
- OpenCV 3.x: 对应 Python 2.7 和 Python 3.x。
- OpenCV 4.x: 对应 Python 3.x。

本书使用多平台适用的Python 3.8.8这个经典版本，非常稳定。

下面简单介绍Python的下载和安装的步骤。

01 Python 3.8.8 版本的安装文件可从官网 www.python.org 下载，我们可以根据自己操作系统的位数选择下载 32 位或者 64 位的安装文件，如图 2-1 所示。

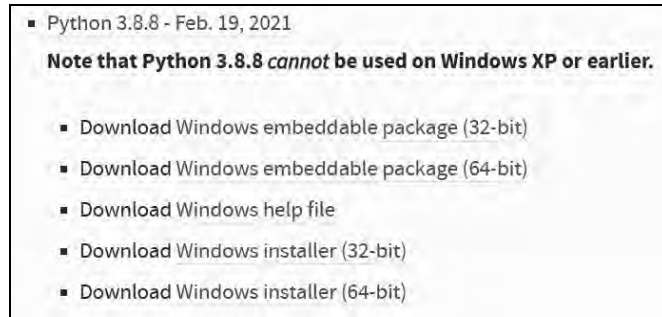


图 2-1

02 有时候官网打开比较慢，建议直接从本书配套资源中的源码目录下的 `somesofts` 文件夹下获取安装文件，文件名是 `python-3.8.8-amd64.exe`，直接双击它就可以开始安装。安装界面如图 2-2 所示。注意，勾选 `Add Python 3.8 to PATH` 复选框，可以把 Python 安装目录加入 `PATH` 环境变量中，再单击 `Install Now` 继续安装。



图 2-2

03 安装成功后，界面如图 2-3 所示。打开一个命令行窗口，输入“python”，就可以查看到 Python 版本（3.8.8）的相关信息了。



图 2-3

学过Python的人都知道，在交互模式的提示符>>>下直接输入代码，按回车键后，就可以立刻得到代码执行结果。试着输入“100+200”，再打印一下“hello, world”，结果如图2-4所示。

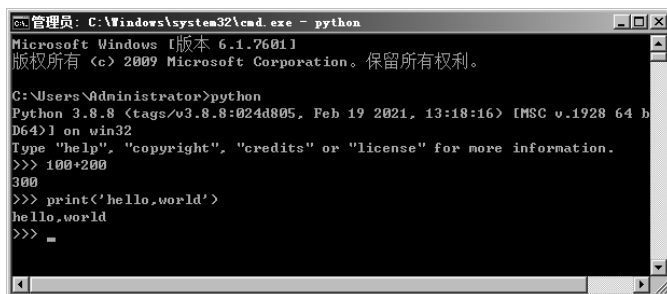


图 2-4

最后，用exit()退出Python提示符。

2.2 在线安装与卸载opencv-python

2.2.1 在线安装 opencv-python

Python安装完成后，就可以搭建OpenCV开发环境。所谓在线安装，意思是安装过程中要联网，边下载边安装。

要用Python开发OpenCV程序，首先要为Python安装配置OpenCV开发包。安装Python相关功能包需要用到pip，它是一个通用的Python包管理工具，提供了对Python包的查找、下载、安装、卸载功能。如果安装的是Python 3.x，就会默认安装pip工具，其位置在Python的安装路径下。比如，笔者计算机上的Python包管理工具pip.exe的路径位于C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Scripts\。这个路径是64位系统的，读者可以查找自己计算机pip.exe的路径。打开命令行窗口，输入“pip”，如果出现很多选项，就说明pip工作正常了，如图2-5所示。

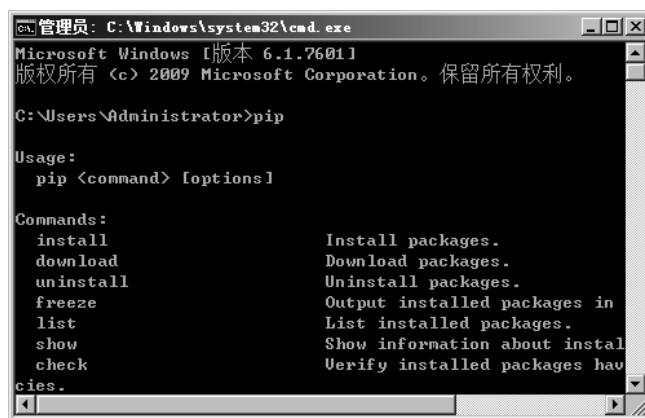


图 2-5

下面利用pip来安装OpenCV开发环境。安装OpenCV开发环境只需要下载NumPy、Matplotlib、opencv-python三个包。其中，NumPy包是Python语言的一个扩展程序库，支持大量的维度数组与矩阵运算，此外也针对数组运算提供了大量的数学函数库。Matplotlib包是Python中类似MATLAB的绘图工具。因为图像处理中有很多直方图统计之类的操作，所以选择了Matplotlib这个Python的第三方包。

首先更新升级setuptools，setuptools也是一个Python配置工具，pip和很多的包管理工具一样，是从国外源下载的，因此速度会比较慢，甚至会安装不了。此时，我们可以指定从国内下载，即在使用pip的时候加上参数 `-i https://pypi.tuna.tsinghua.edu.cn/simple`，这样就会从清华镜像库去安装setuptools库。命令如下：

```
pip install --upgrade setuptools -i https://pypi.tuna.tsinghua.edu.cn/simple
```

这个包下载安装比较快，如果出现错误提示“`mysql-connector-python 8.0.25 requires protobuf>=3.0.0, which is not installed`”，可以把protobuf安装一下，安装命令是“`pip3 install protobuf`”，然后再次运行升级命令。

接下来安装NumPy和Matplotlib库，在命令行下输入如下命令：

```
pip install matplotlib -i https://pypi.tuna.tsinghua.edu.cn/simple
```

这一步下载需要花点时间，最终下载完成如图2-6所示。

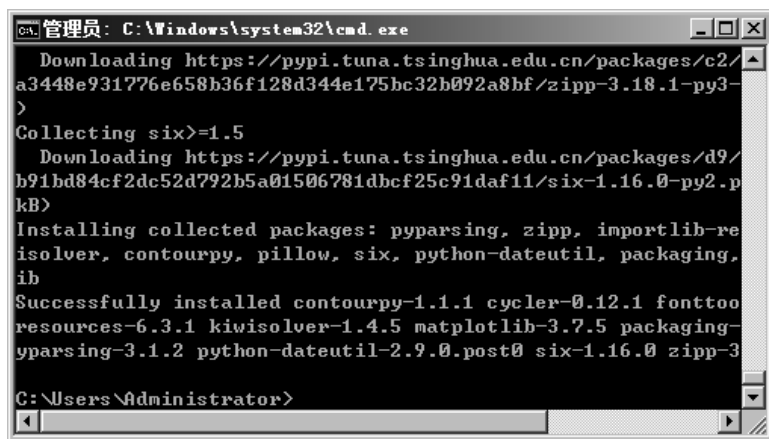


图 2-6

最后安装opencv-python，本书用的OpenCV版本是4.10，我们可以指定版本号来安装opencv-python，在命令行下输入如下命令：

```
pip install opencv-python==4.10.0.84 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

注意，上面命令中版本号4.10.0.84前面有2个等于号（==）。运行该命令后稍等片刻，opencv-python安装完成，如图2-7所示。

此时一个名为cv2的文件夹已经生成在以下路径：

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Lib\site-packages\
```

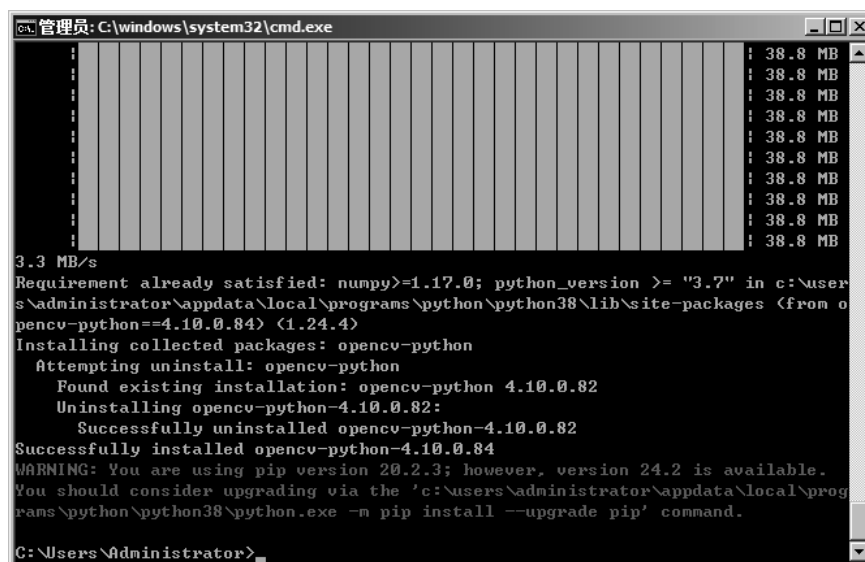


图 2-7

cv2文件夹存放的是OpenCV编程所需要的内容。如果要查看当前计算机上安装的OpenCV版本，可以运行命令“pip show opencv-python”。

至此，基于Python的命令行OpenCV开发环境就建立起来了。下面开始用Python开发第一个OpenCV程序。此时没有安装IDE集成开发环境，可以暂时使用记事本编辑这个程序。所谓命令行OpenCV开发环境，就是OpenCV程序需要在命令行窗口下用命令来运行。这种方式效率不高，一线企业级开发一般使用集成开发环境。

【例 2.1】 第一个 OpenCV 程序

```
#coding=gbk
#导入 cv 模块
import cv2 as cv
#读取图像，支持 BMP、JPG、PNG、TIFF 等常用格式
img = cv.imread("opencv-logo2.png")
cv.imshow("Hello,python opencv",img) #显示窗口
#等待按键
cv.waitKey(0)
#释放窗口
cv.destroyAllWindows()
```

首先利用import语句导入模块cv2（OpenCV是由很多个模块组成的，cv2是OpenCV中的一个基本模块）。使用as语法之后，只能通过as后面的名字来访问导入的模块，因此后面代码要用cv来代表cv2。

模块导入后，就可以使用cv2里的函数了。我们先利用函数cv.imread读取一个文件opencv-logo2.png，该文件位于工程目录下。cv.imread也可以传入一个绝对路径，比如d:\opencv-logo2.png。接着用函数imshow把图片显示在窗口中。函数waitKey用于等待用户按键，如果用户不按键，则图片窗口会一直显示，也就是函数waitKey一直处于阻塞状态，直到用户

按键才返回。waitKey结束后调用函数destroyAllWindows销毁所有窗口，这里只有一个窗口，就是我们用namedWindow函数创建的窗口。

把上述代码在记事本中输入后，在某个路径下保存为2.1.py。打开命令行窗口并定位到2.1.py所在的目录，然后执行如下命令：

```
python 2.1.py
```

注意 在上面命令中，python后面有一个空格，而且图片文件opencv-logo2.png要和2.1.py在同一个目录下。

最终运行结果如图2-8所示。

运行成功，说明我们的opencv-python开发环境搭建成功了。下面再看一个稍微复杂一点例子。



图 2-8

【例 2.2】 把两幅图片混合后输出

```
import cv2 as cv                #导入 cv 模块
import numpy as np              #导入 numpy 模块
import sys
alpha = 0.5
print("线性混合")
input=float(input('* 输入第一幅图片的权重 alpha [0.0-1.0]:'))
print(input)
if 0 <= input <= 1:            #判断输入合法性
    alpha = input
src1 = cv.imread("p1.jpg");     #读取第一幅图片
src2 = cv.imdecode(np.fromfile("山水.jpg",dtype=np.uint8),-1)      #读取图片
if src1 is None:                #判断 p1.jpg 是否读取成功
    sys.exit("Could not read the p1.jpg.")      #如果读取为空，则退出程序
if src2 is None:                #判断山水.jpg 是否读取成功
    sys.exit("Could not read the 山水.jpg.")    #如果读取为空，则退出程序
beta = (1.0 - alpha);
dst=cv.addWeighted(src1, alpha, src2, beta, 0.0, 0)      #将图 1 与图 2 线性混合
cv.imshow("result",dst)        #显示混合后的图片
cv.waitKey(0)                  #等待按键
```

NumPy模块包含大量的数学运算的函数，包括三角函数、算术运算的函数、复数处理函数等。上述代码的功能是将图片p1.jpg和山水.jpg进行混合，它们的大小必须一样。这两幅图片目前都保存在工程源码目录下。其中，imread函数用来读取图片，由于imread不支持中文文件名，因此通过np.fromfile来读取山水.jpg。fromfile可以支持中文文件名，这个函数的用法会在后面章节详细讲解，这里只需要了解即可。

addWeighted函数是将两幅相同大小、相同类型的图片进行融合，第二个参数alpha表示第一幅图片所占的权重，第四个参数beta表示第二幅图片所占的权重。权重越大，图片显示得越多，比如设置alpha为0.9（见图2-9），则主要显示第一幅图片。

运行工程，结果如图2-10所示，可以看到设置第一幅图片的权重为0.9后，第二幅图片就淡了很多。



图 2-9



图 2-10

至此，基于Python的OpenCV的开发环境就搭建起来了。

2.2.2 卸载 opencv-python

这里讲解如何卸载opencv-python，不是为了马上卸载opencv-python，而是为了以后升级opencv-python时，可以先把旧的opencv-python卸载掉再安装新版的opencv-python。

打开命令行窗口，输入命令“pip uninstall opencv-python”并按回车键，此时出现让我们确认是否删除的提示，如图2-11所示。

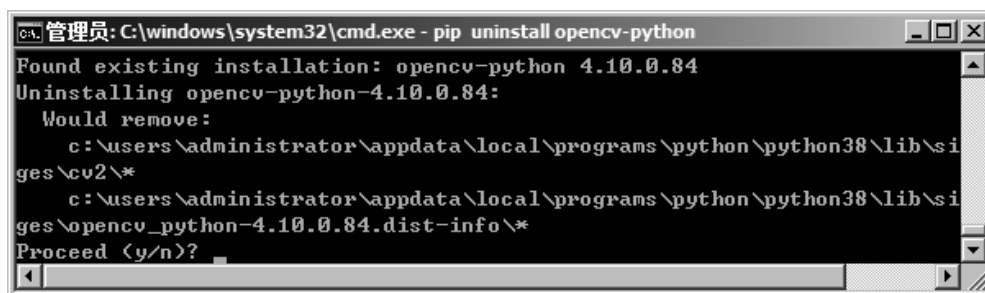


图 2-11

按y键开始卸载。然后到下面路径查看一下：

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Lib\site-packages\
```

可以发现cv2文件没有了，说明卸载成功。

2.3 离线安装opencv-python

考虑到不少读者的工作计算机无法连外网，还是要讲一下离线安装opencv-python。基本过程就是先完整下载opencv-python离线安装包，然后就可以断网在本地安装了。

2.3.1 下载离线版 opencv-python

离线安装包的下载网址是<https://pypi.org/project/opencv-python/#files>。该网站打开比较慢，下载下来的文件是opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl，读者可以直接在本书配套资源中的somesofts目录下查找该文件。

whl文件是一个压缩文件，里面包含了py文件，以及经过编译的pyd文件，方便在机器上进行Python模块的安装。cp37代表需要的Python版本是3.7，我们如果安装的是Python 3.8，这影响不大，用起来基本一样。

2.3.2 离线安装 NumPy 和 Matplotlib 库

和在线安装opencv_python一样，离线安装之前也要先安装NumPy和Matplotlib库。这两个库可以到网站<https://pypi.org/>上去搜索，然后根据不同的操作系统和已安装的Python版本去下载。这里下载下来的NumPy文件是numpy-1.23.2-cp38-cp38-win_amd64.whl，下载下来的Matplotlib文件是matplotlib-3.4.3-cp38-cp38-win_amd64。笔者也把这两个文件和源码目录放到配套资源中了，免得读者去搜索下载了。

打开命令行窗口，进入numpy-1.23.2-cp38-cp38-win_amd64.whl文件所在目录，输入如下命令：

```
pip install whl numpy-1.23.2-cp38-cp38-win_amd64.whl
```

稍等片刻，安装完成。再进入matplotlib-3.4.3-cp38-cp38-win_amd64.whl文件所在目录，输入命令如下：

```
pip install whl matplotlib-3.4.3-cp38-cp38-win_amd64.whl
```

稍等片刻，安装完成，如图2-12所示。

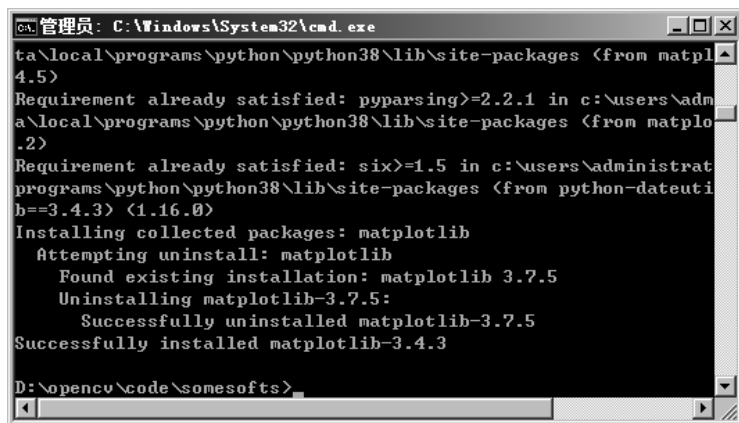


图 2-12

2.3.3 离线安装 opencv-python

前面我们已经下载好了opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl这个离线安装包，现在可以用pip命令来安装了。但在安装之前，如果前面已经在线安装过opencv-python，

则必须先卸载再安装。卸载命令是“`pip uninstall opencv-python`”。卸载后就可以通过whl文件安装了。

打开命令行窗口，进入`opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl`所在的目录，然后输入如下命令：

```
pip install whl opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl
```

稍等片刻，安装完成，如图2-13所示。

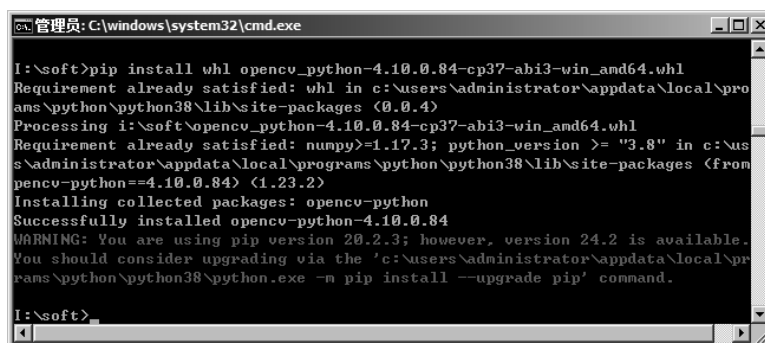


图 2-13

此时，在以下路径下可以看到有一些cv2文件夹了：

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Lib\site-packages
```

在命令行下输入`pip list`，可以看到结果中已经有`opencv-python`了，如下所示：

```
opencv-python      4.10.0.84
```

除此之外，还可以在命令行窗口中输入`python`，然后在提示符下输入`import cv2`，如果未提示任何信息，则说明OpenCV安装成功了。

至此，通过离线安装包安装`python-opencv`成功完成。

2.4 使用集成开发环境PyCharm

和其他语言类似，Python程序可以使用Windows自带的命令行窗口通过命令来运行，但是这种方式对于较为复杂的程序工程来说，容易混淆相互之间的层级和交互文件。因此，在编写程序工程时，建议使用专用的集成开发环境PyCharm。

2.4.1 PyCharm 的下载和安装

读者可以直接到PyCharm官网下载最新版，也可以使用笔者提供的安装包。在配套资源的`somesofts`子目录下找到安装包文件`pycharm-community-2022.1.1.exe`，该版本是目前主流版本，而且适用面广（支持Windows 7、Windows 10、Windows 11）。

直接双击它开始安装，安装过程比较简单。采用默认配置安装即可。如果想在桌面上生成 PyCharm 程序的快捷方式，可以在“安装选项”对话框上对“PyCharm Community Edition”前的复选框打勾，如图2-14所示。稍等片刻，安装完成，如图2-15所示。

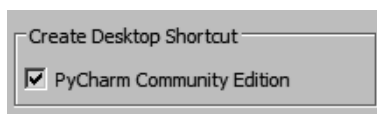


图 2-14



图 2-15

最后单击“Finish”按钮，确认PyCharm安装完成。现在马上趁热打铁，创建一个简单的PyCharm项目并运行，从而测试PyCharm是否安装成功。

【例 2.3】 第一个 PyCharm 项目

(1) 单击桌面上新生成的  图标启动PyCharm，如果是第一次启动会出现Import PyCharm Settings对话框，意思是让我们导入PyCharm这个软件的配置，一般建议选择Do not import settings，即不导入配置，也就是由PyCharm自动指定即可。之后单击OK按钮，完成初始化设定。

(2) 此时出现欢迎对话框，如图2-16所示。该对话框常用的按钮有两个，一个是New Project，用于创建新工程；另外一个Open，用于打开一个已经存在的工程。而Get From VCS的意思是从版本控制工具VCS上面获取源码，这个初学者可以不用关注，一般企业里稍大的项目才会用到版本控制。

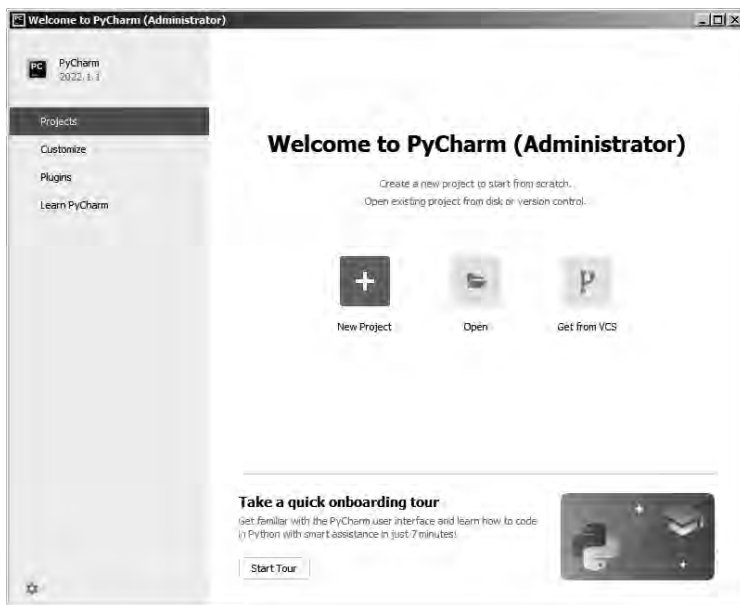


图 2-16

(3) 单击New Project按钮来创建新工程，此时出现New Project对话框，在该对话框最上方的Location旁输入新工程的存储路径，比如D:\ex\mycv\pythonProject，并勾选Inherit global site-packages复选框，其他保持默认即可，如图2-17所示。

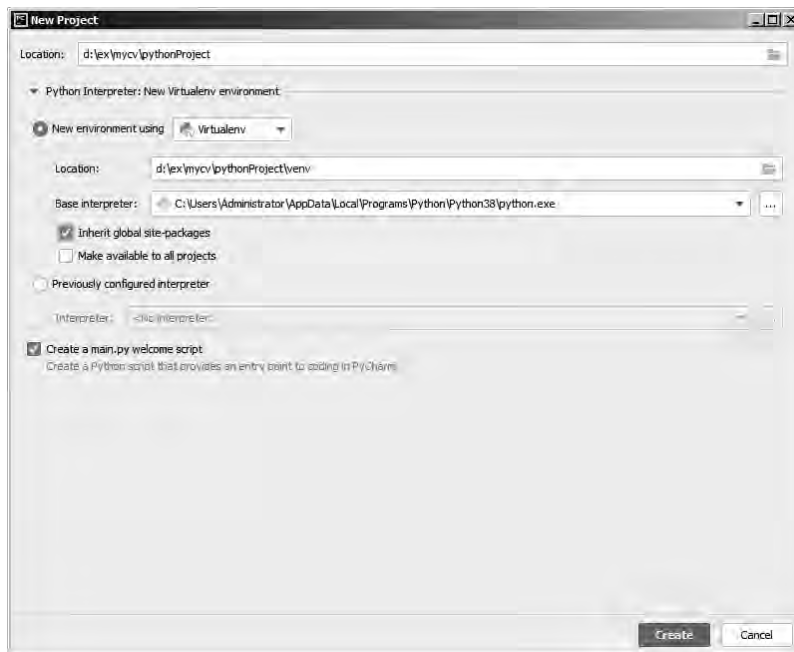


图 2-17

(4) 单击Create按钮，PyCharm将创建一个简单的新项目，并生成一个main.py源码文件。这个文件里面已经有几行代码了，如下所示：

```
def print_hi(name):
    # Use a breakpoint in the code line below to debug your script.
    print(f'Hi, {name}') # Press Ctrl+F8 to toggle the breakpoint.

# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    print_hi('PyCharm')
```

代码很简单，就是两个函数：一个是main主函数，里面调用了自定义函数print_hi；另外一个是为print_hi的自定义打印函数，里面调用了库函数print来输出字符串。

(5) 让我们马上来运行这个文件。单击菜单栏中的Run→Run...或直接按快捷键Ctrl+F5，或者在左边的Project视图中右击main.py文件名，在弹出的快捷菜单中选择Run 'main'，还可以直接单击PyCharm右上角的三角箭头图标按钮，如图2-18所示。



图 2-18

代码运行后，如果在下方输出窗口中出现“Hi, PyCharm”，就表示PyCharm工作正常。现在我们学会了怎么新建一个项目，下面打开一个已经存在的项目，以此来加深PyCharm的使用体会。

【例 2.4】 打开现有项目

(1) 把例2.3的项目文件夹pythonProject复制一份，然后粘贴到磁盘某个路径下。笔者这里是把项目文件夹pythonProject剪切到路径D:\ex\mycvNew\下。注意是剪切，如果不小心用了复制，那就把路径d:\ex\mycv\下的pythonProject目录删掉。笔者故意用了一个新路径，目的是演示使用PyCharm打开一个已经存在的项目。

(2) 打开PyCharm，在Welcome to Charm对话框上单击Open按钮通过Open按钮可以打开一个文件或一个项目，这里准备打开一个已经存在的项目。然后在Open File or Project对话框上选择项目文件夹，如图2-19所示。

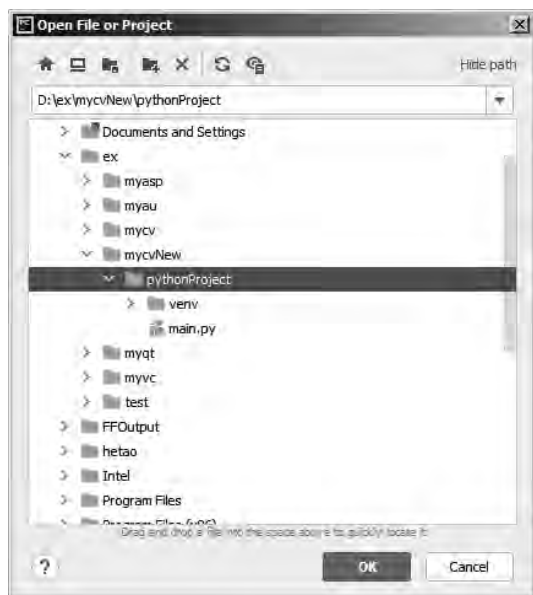


图 2-19

接着单击OK按钮。此时会出现对话框，询问是否在当前窗口或新的窗口中打开项目，如图2-20所示。

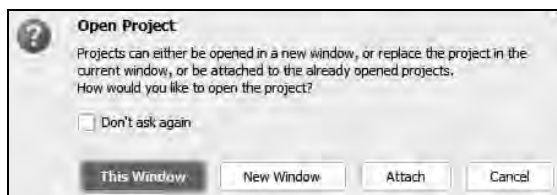


图 2-20

我们直接单击This Windows按钮即可。随后，PyCharm将加载项目pythonProject，并自动打开main.py文件。然而，此时直接运行项目会报错，提示找不到Python解释器。这是因为这个项目最初是在d:\ex\mycv\下创建的，所以使用的Python解释器的路径是：

```
d:\ex\mycv\pythonProject\venv\Scripts\
```

由于d:\ex\mycv\下的pythonProject目录被删除了，而且项目文件夹移动到D:\ex\mycvNew\下，PyCharm依旧去d:\ex\mycv\pythonProject\venv\Scripts\下找解释器，当然找不到了。那怎么办呢？当然是要重新为项目配置解释器。

(3) 开始配置解释器。在PyCharm中，单击菜单File→Setting，打开Settings窗口，然后在Settings窗口的左边单击Project: pythonProject，并选择Python Interpreter，此时可以看到右边Python Interpreter旁出现 “[Invalid] Python 3.8(pythonProject)....” 这样的提示，意思就是d:\ex\mycv\pythonProject\venv\Scripts\python.exe这个解释器已经无效了，如图2-21所示。

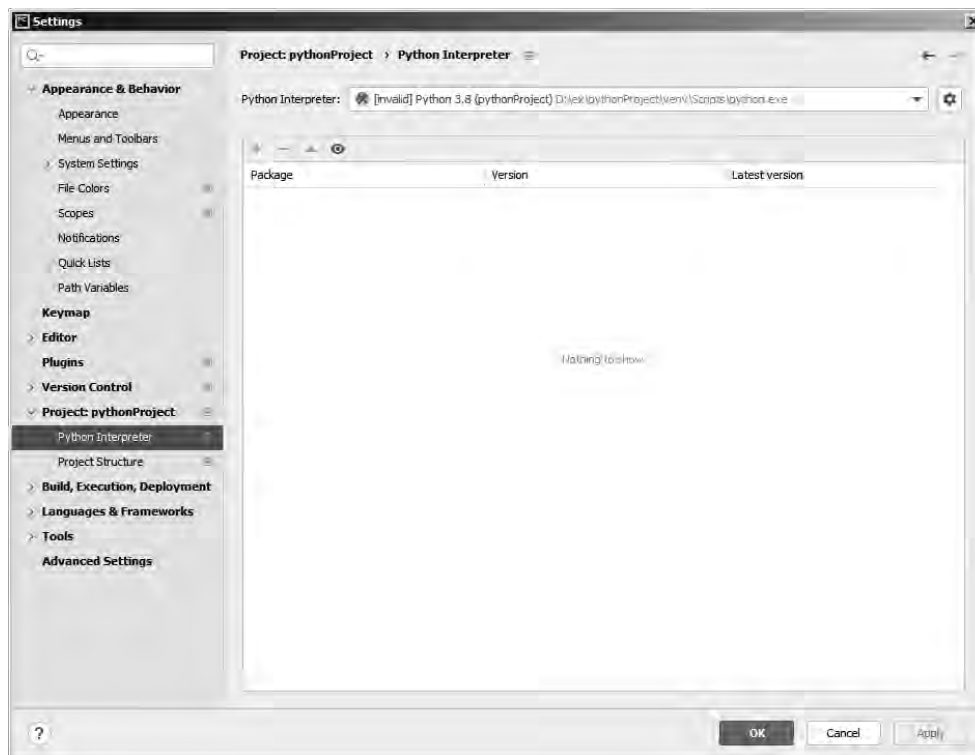


图 2-21

现在单击Python Interpreter右方的下拉箭头，在下拉菜单中选择Show All...，此时出现Python Interpreters对话框。在该对话框中，选中那个无效的解释器，如图2-22所示。



图 2-22

然后单击左上角的“-”图标或者直接按快捷键Alt+Delete来删除这个无效的解释器。删除后，再单击左上角的“+”图标或者直接按快捷键Alt+Insert来准备新增解释器，此时会出现Add Python Interpreters对话框。在该对话框中，一般会探测到新项目文件夹中解释器路径，并显示在Existing environment中，如图2-23所示。

直接单击OK按钮，回到Python Interpreters对话框，再单击该对话框上的OK按钮。此时我们在Settings对话框上可以看到Python解释器被成功添加了，如图2-24所示。

单击OK按钮关闭Settings对话框。然后在PyCharm中运行该项目，如果出现Edit Configuration对话框，则只需在Python Interpreter下拉菜单中选择刚刚添加的解释器即可，如图2-25所示。

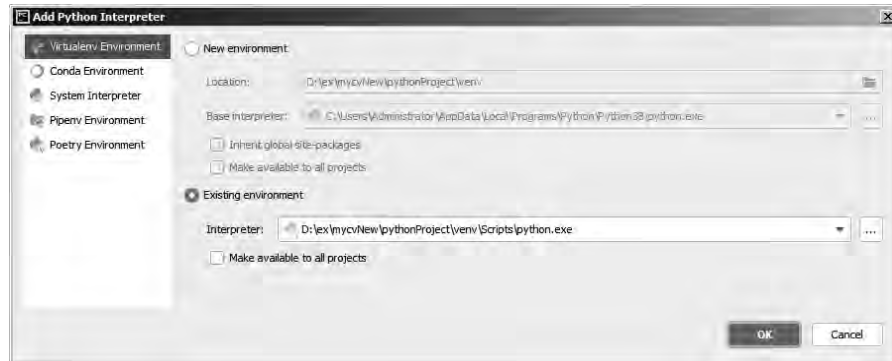


图 2-23

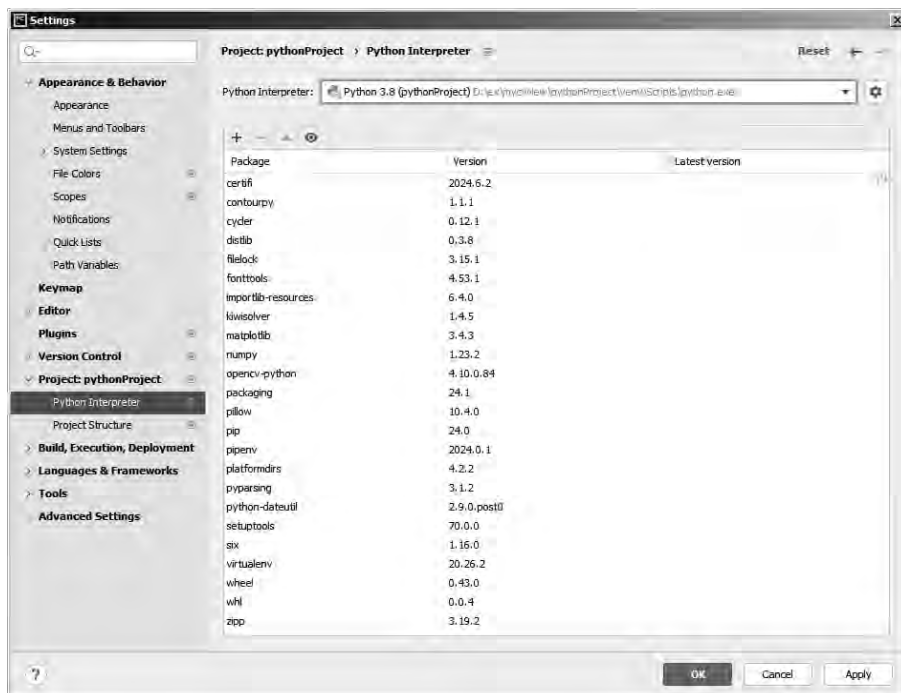


图 2-24

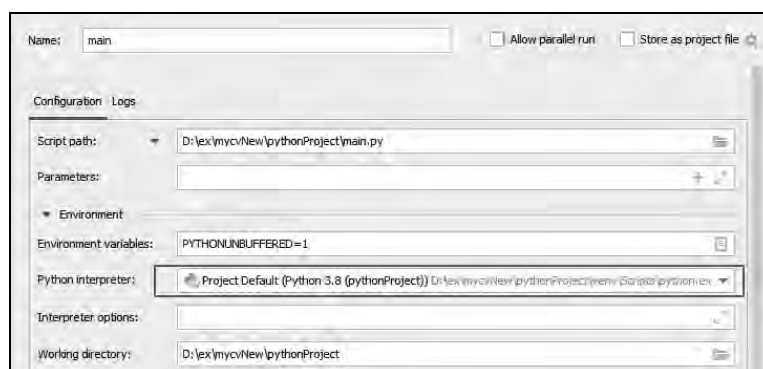


图 2-25

随后可以正确运行新工程，运行结果如下：

```
Hi, PyCharm
```

2.4.2 了解 PyCharm 的虚拟环境

在图2-17中，在New environment using旁有个默认选项Virtualenv，这是什么意思呢？从字面意思看就是“新环境使用虚拟环境”。什么意思呢？

应用程序有时需要某个特定版本的库，因为它需要一个特定的bug已得到修复的库，或者它是使用一个过时版本的库的接口编写的。这就意味着无法通过一个Python来满足每个应用程序的要求。例如，应用程序A需要一个特定模块的1.0版本，但是应用程序B需要该模块的2.0版本，这两个应用程序的要求是冲突的，安装版本1.0或者版本2.0将会导致其中一个应用程序不能运行。

这个问题的解决方案就是为不同的应用各自创建一个虚拟环境（通常简称为 virtualenv），包含一个特定版本的Python，以及一些附加包的独立的目录树。

应用程序A可以有自己的虚拟环境，其中安装了特定模块的1.0版本。而应用程序B拥有另外一个安装了特定模块2.0版本的虚拟环境。当应用程序B需要一个库升级到3.0版本时，也不会影响到应用程序A的环境。

如果直接在PyCharm中创建一个项目而不创建虚拟环境，那么安装的第三方包都会安装到系统Python解释器的site-packages文件夹下，比如：

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Lib\site-packages\
```

创建越多的项目，安装的库就越多。当我们新建一个项目时，必定会把site-packages下的所有库都导进来，而一些库这个项目根本就不需要，但是又不能删除（因为别的项目在用），这时就需要虚拟环境了。如果只创建一两个项目用于Python入门，那么用不用虚拟环境都不影响。了解了虚拟环境的背景知识，我们再来看一下Creat Project对话框，如图2-26所示。



图 2-26

New environment using 下拉菜单中有 4 个选项，其中 3 个是虚拟环境，分别是 Virtualenv、Pipenv 和 Conda，如图 2-27 所示。下面简要介绍一下前面两种。

1) Virtualenv

Virtualenv是本书默认使用的虚拟环境方式。通过Virtualenv创建的虚拟目录放在本项目下的venv文件夹中。比如，假设项目地址是E:\PycharmProjects\pythonProject，则虚拟环境的地址就是E:\PycharmProjects\pythonProject\venv，虚拟环境中库的地址就是E:\PycharmProjects\pythonProject\venv\Lib\site-packages。

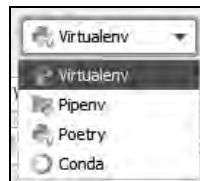


图 2-27

Base Interpreter（基本解释器）就是系统Python解释器，也就是我们前面安装的Python软件，并配置好了环境变量。一般系统Python解释器的第三方库都在site-packages目录下，比如笔者的：

```
C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Lib\site-packages\
```

Virtualenv一般配合requirements.txt文件对项目的依赖库进行管理。requirements.txt文件记录了本虚拟环境下的依赖库及其版本号，其生成方式如下：

打开Pycharm，按照例2.3的方式新建一个项目，然后单击菜单Tools → Sync Python Requirements，此时出现Sync Python requirements对话框，如图2-28所示。

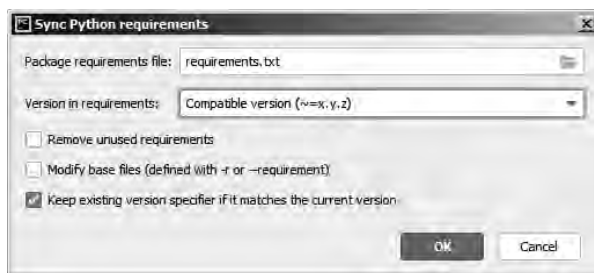


图 2-28

直接单击OK按钮，根目录下自动生成一个空的requirements.txt 文件。此时打开 PyCharm 左下角的 Terminal（终端），在终端窗口的命令行旁输入如下命令后按回车键：

```
pip freeze > requirements.txt
```

此时requirements.txt中生成的内容如下：

```
contourpy==1.1.1
cycler==0.12.1
fonttools==4.53.0
importlib-resources==6.4.0
kiwisolver==1.4.5
matplotlib==3.7.5
numpy==1.24.4
opencv-python==4.10.0.84
packaging==24.1
pillow==10.3.0
pyparsing==3.1.2
python-dateutil==2.9.0.post0
```

```
six==1.16.0
zipp==3.19.2
```

这样的结构让人一目了然，且方便项目移植。当复制一个含有requirements.txt文件的项目时，可以在终端窗口中输入以下命令来安装所有的依赖库：

```
pip install -r requirements.txt
```

2) Pipenv

Pipenv是一种工具，提供了为Python项目创建虚拟环境所需的所有必要方法。在安装或卸载软件包时，它会通过Pipfile文件自动管理项目软件包。

Pipenv还会生成Pipfile.lock文件，该文件用于生成确定性构建并创建工作环境的快照。当项目要求和软件包版本很重要时，这对于安全敏感型的应用部署特别有用。

如果当前版本的Pycharm没有内置Pipenv，则需要安装。安装步骤如下：

01 单击开始菜单，输入cmd，打开命令行窗口，运行以下命令以确保系统中已安装pip：

```
pip --version
```

02 pipenv通过运行以下命令进行安装：

```
pip install --user pipenv -i https://pypi.tuna.tsinghua.edu.cn/simple
```

03 安装成功后可以查看pipenv.exe所在路径。在命令行窗口输入以下命令：

```
py -m site --user-site
```

返回的是库路径，比如：

```
C:\Users\Administrator\AppData\Roaming\Python\Python38\site-packages
```

将此路径最后的site-packages替换为Scripts，则变成：

```
C:\Users\Administrator\AppData\Roaming\Python\Python38\Scripts
```

这个路径就是pipenv.exe所在的目录。

通过Pipenv创建的项目，其虚拟环境并不在本项目的目录下，而是在“C:\Users\当前用户名\virtualenvs”文件夹下。在Pipenv虚拟环境中不用requirements.txt，Pipfile是Pipenv虚拟环境用于管理项目依赖项的专用文件，该文件对于使用Pipenv是必不可少的。当为新项目或现有项目创建Pipenv环境时，会自动生成Pipfile。

2.4.3 在 PyCharm 下开发 OpenCV 程序

前面PyCharm开发的程序都是纯Python程序，并没有涉及OpenCV的代码。现在我们要在PyCharm下开发OpenCV代码。由于我们已经正确安装了Matplotlib、NumPy、opencv-python等开发包，因此，只要PyCharm能正确找到Python解释器，那么位于路径C:\Users\Administrator\AppData\Local\Programs\Python\Python38\Lib\site-packages\下的库就都可以被找到，则该路径下的cv2目录也可以被PyCharm感知，这样调用OpenCV库中的函数就不会报错了。如果把cv2改个名称，PyCharm也能马上感知到，并报错，无法运行OpenCV相关代码。

【例 2.5】 第一个 PyCharm 下的 OpenCV 程序

(1) 打开 PyCharm，新建一个项目，项目目录是 D:\ex\mycv\，项目名称保持默认 pythonProject。注意，不要忘记勾选 Inherit global site-packages 复选框，以后示例中新建项目时都要勾选它，但不再提醒。

(2) 在 main.py 中输入如下代码：

```
import cv2 as cv

if __name__ == '__main__':
    # 读取图像，支持 bmp、jpg、png、tiff 等常用格式
    img = cv.imread("tree.png")
    cv.imshow("Hello,tree", img) # 显示窗口
    cv.waitKey(0) # 等待按键
    cv.destroyAllWindows() # 释放窗口
```

这些代码和例 2.1 中的代码几乎一样。注意把图片文件 tree.png 放到项目文件夹 pythonProject 下。

(3) 按快捷键 Ctrl+Shift+F10 运行代码，运行结果如图 2-29 所示。

是不是感觉很简单。其实，只需通过 pip 正确安装好 OpenCV，也就是 site-packages 目录下存在 cv2 目录即可。不信的话，可以把 cv2 改个名称再运行本示例，马上报错！



图 2-29

2.4.4 调试 Python 程序

和开发其他程序一样，Python 程序有时也需要调试，比如设置断点、运行到断点处、单步执行、监视某个变量等。

在 PyCharm 中单步调试 Python 程序非常简单，只需设置好断点，然后按快捷键 Shift+F9 开始调试运行，就会执行到断点处，然后按快捷键 Shift+F8 开始单步执行（按快捷键 Shift+F7 也可以单步执行，并且遇到函数还能进入函数里面执行），如图 2-30 所示。

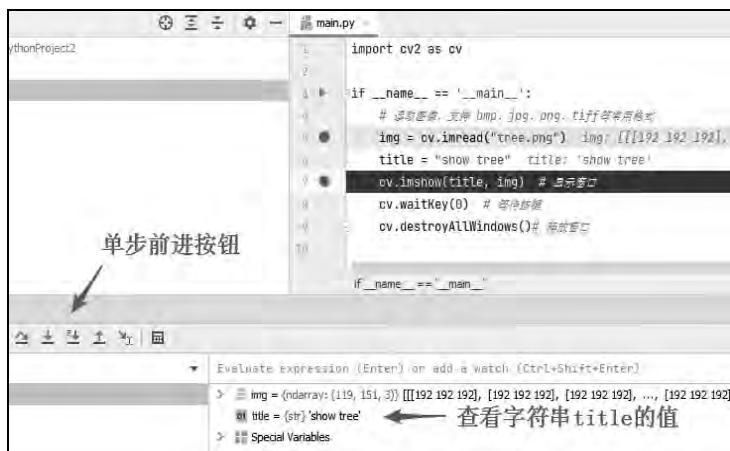


图 2-30

在窗口下方的Debug视图中，我们直接可以看到变量a的相关信息。通常，设置断点、单步执行、监视变量这三大手段对于调试基本够用了。

另外，在运行程序时，如果出现了语法错误，可以到PyCharm下方的“运行”窗口中查看错误说明，如图2-31所示。

图2-31中显示，第6行（line 6）出错了。如果我们需要定位这一行，可以单击错误提示信息中带下画线的部分，编辑器即可跳转到相应的行。

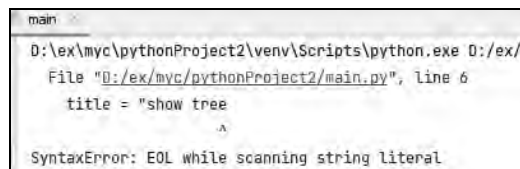


图 2-31

2.5 测试一下NumPy的数学函数

本节测试一下NumPy中的数学函数，看它是否能工作正常。NumPy支持大量的维度数组与矩阵运算，并针对数组运算提供了大量的数学函数库，常用的数学函数如下：

- `round(a, decimals=0, out=None)`: 将小数 a 四舍五入到给定的小数位数。
- `floor(a)`: 取比小数 a 小的最大的整数，即向下取整。
- `ceil(a)`: 取比小数 a 大的最小的整数，即向上取整。

其中，a可以是一个数字，也可以是一个数组。

【例 2.6】 测试取整的数学函数

```
import cv2 as cv #导入 cv 模块
import numpy as np
a = np.array([1.0, 5.55, 123, 0.567, 25.532])
print("np.round(2.6) : " , np.round(2.6))
print("np.round(a,1):", np.round(a,1))
print("np.round(2.8) : " , np.round(2.8))
print("cvFloor(2.5) : " , np.floor(2.5))
print("cvFloor(2.6) : " , np.floor(2.6))
print("cvCeil(2.5) : " , np.ceil(2.5))
print("cvCeil(2.6) : " , np.ceil(2.6))
```

在上述代码中，分别测试了round、floor和ceil数学函数的简单使用。运行工程，结果如图2-32所示。

```
np.round(2.6) : 3.0
np.round(a,1): [ 1.   5.6 123.   0.6 25.5]
np.round(2.8) : 3.0
cvFloor(2.5) : 2.0
cvFloor(2.6) : 2.0
cvCeil(2.5) : 3.0
cvCeil(2.6) : 3.0

Process finished with exit code 0
```

图 2-32