数字图像处理基础工具

Python 是一种解释型、面向对象、动态数据类型的高级程序设计语言，其语法简单易懂，受到许多开发者喜爱。使用 Python 处理图像时可以借助许多好用的工具包，主流图像处理工具包有：OpenCV、SciKit-Image、Pillow、Numpy 和 Matplotlib 等。

1 Python 介绍与安装

1.1 Python 介绍

Python 是由 Guido van Rossum 在八十年代末和九十年代初，在荷兰国家数学和计算机科学研究所设计出来的。Python 本身也是由诸多其他语言发展而来的,这包括 ABC、Modula-3、C、C++、Algol-68、SmallTalk、Unix shell 和其他的脚本语言等等。像 Perl 语言一样，Python 源代码同样遵循 GPL(GNU General Public License)协议。现在 Python 是由一个核心开发团队在维护，Guido van Rossum 仍然占据着至关重要的作用，指导其进展。Python 2.7 被确定为最后一个 Python 2.x 版本，它除了支持 Python 2.x 语法外，还支持部分 Python 3.1 语法。目前 Python2 的最新版本为 2.7，官方不在更新 Python2，教学中推荐使用 Python3。

1.2 Python 安装

Python3 目前已经更新到 3.10.2，本示例使用 3.9 的版本，其支持的包更多一些。Python3 的安装方法主流有两个，一个是使用官方 Python3 原生版本，第二个是使用 Anaconda 创建 Python3 虚拟环境。

1.2.1 官方 Python3 原生版本

1）下载安装程序

打开 Python 官网连接 https://www.python.org/。

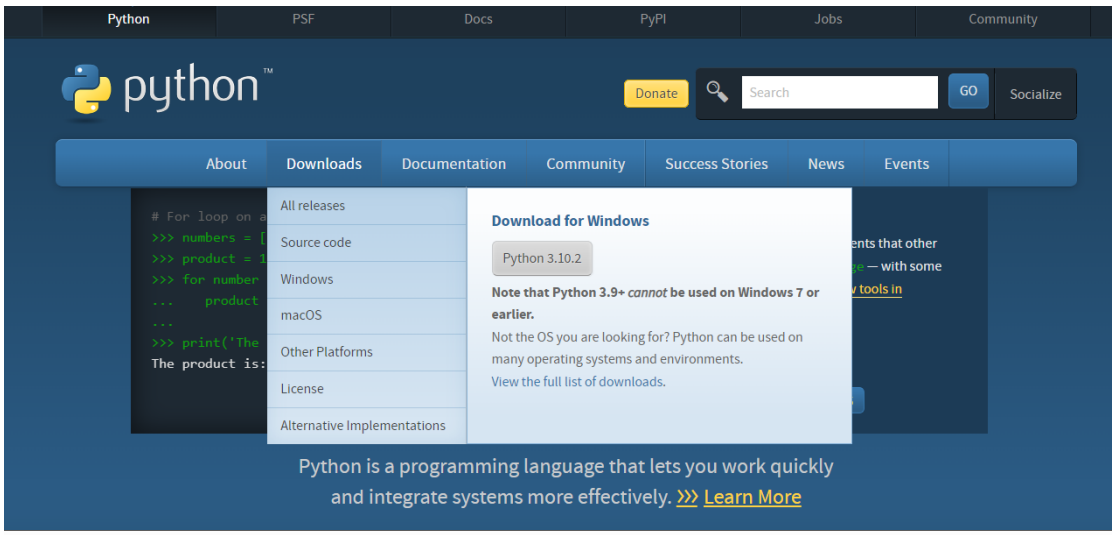


图 1.1 Python 官网

如图 1.1 所示，在 Downloads 栏目下选择对应平台下的安装程序，平台包括 Windows、macO 和 Linux。其中 Windows 和 macOS 平台下载对应安装程序即可。Linux 平台需要通过编译源码的方式安装。

2）安装

这里以 Windows 为例介绍安装步骤。点击对应平台，这里选择 Windows。

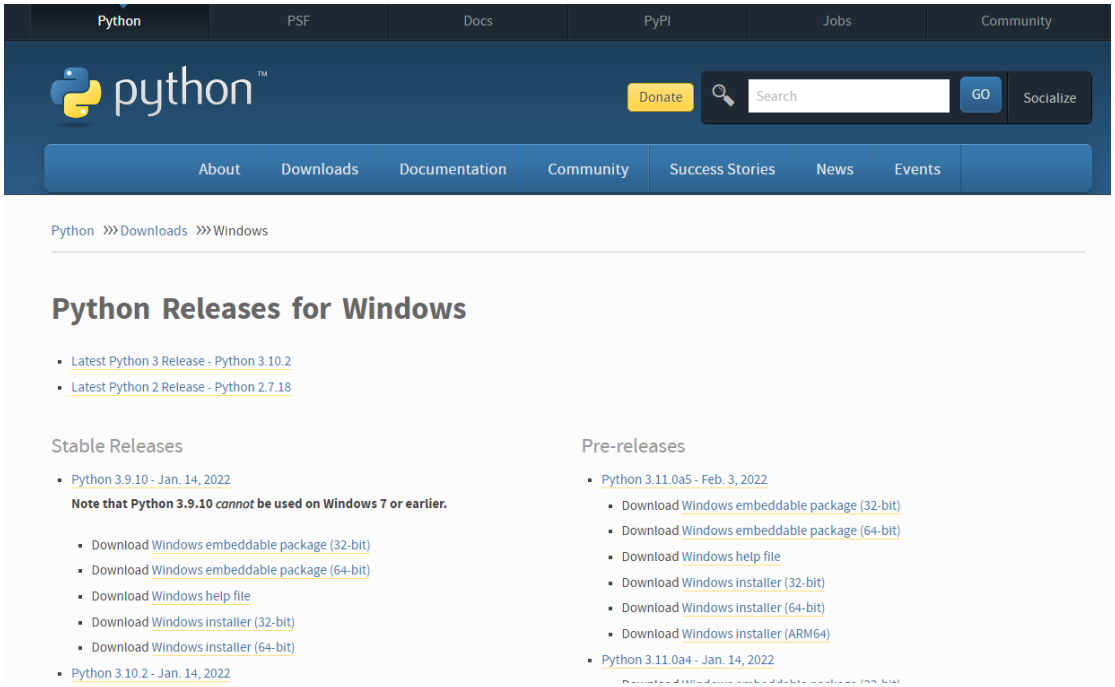
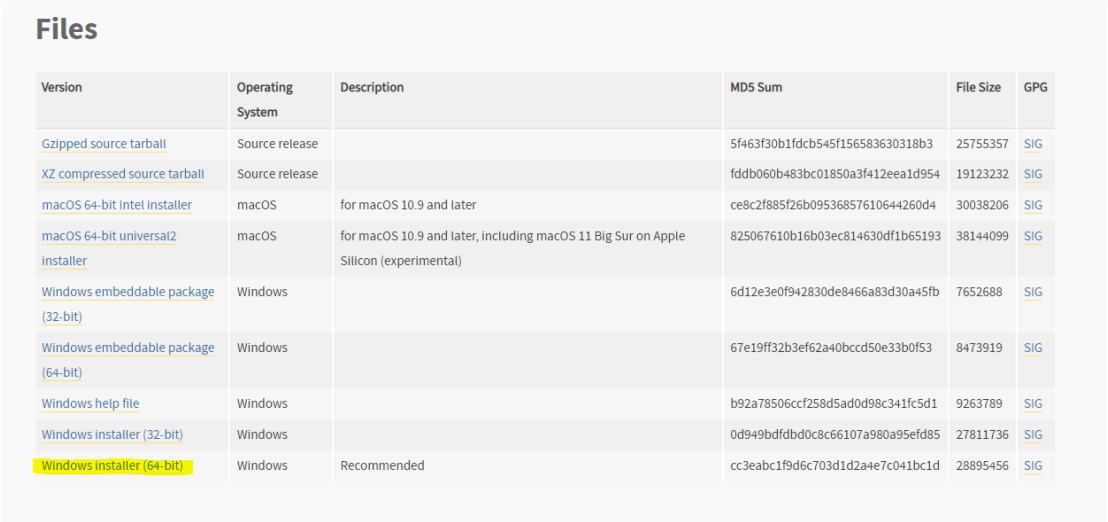


图 1.2　Windows 的 Python 版本

如图 1.2 所示，页面列出了 Python 的最新版本和历史版本的安装包，可以选择下载最新的版本 3.10，也可以选择历史版本（3.7/3.8/3.9）。点击选择的版本，例如 Python3.9.7。



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		5f463f30b1fdcb545f156583630318b3	25755357	SIG
XZ compressed source tarball	Source release		fddb060b483bc01850a3f412eea1d954	19123232	SIG
macOS 64-bit Intel installer	macOS	for macOS 10.9 and later	ce8c2f885f26b09536857610644260d4	30038206	SIG
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later, including macOS 11 Big Sur on Apple Silicon (experimental)	825067610b16b03ec814630df1b65193	38144099	SIG
Windows embeddable package (32-bit)	Windows		6d12e3e0f942830de8466a83d30a45fb	7652688	SIG
Windows embeddable package (64-bit)	Windows		67e19ff32b3ef62a40bccd50e33b0f53	8473919	SIG
Windows help file	Windows		b92a78506ccf258d5ad0d98c341fc5d1	9263789	SIG
Windows installer (32-bit)	Windows		0d949bdfdbd0c8c66107a980a95efd85	27811736	SIG
Windows installer (64-bit)	Windows	Recommended	cc3eabc1f9d6c703d1d2a4e7c041bc1d	28895456	SIG

图 1.3　Python3.9.7 选择列表

如图 1.3 所示，根据电脑位数选择安装文件，如 Windows installer(64-bit)。一般新式电脑的位数都为 64，老电脑会有 32 位的。点击下载安装程序到文件夹中。下载后，双击下载包，进入 Python 安装向导，安装非常简单，你只需要使用默认的设置一直点击"下一步"直到安装完成即可。

1.2.2 Anaconda 虚拟环境

Anaconda 里面集成了很多关于 python 科学计算的第三方库，主要是安装方便，而 python 是一个编译器，如果不使用 anaconda，那么安装起来会比较痛苦，各个库之间的依赖性就很难连接的很好。Anaconda 可以进行虚拟环境的隔离，可以做到多种 python 版本管理，使用起来比较方便。

1）下载 Anaconda 安装包

官网下载 Anaconda，打开 https://www.anaconda.com/products/individual，下载对应平台和位数的安装包。

图 1.4　Anaconda 安装包下载

如图 1.4 所示，下载 Windows 下 64 位的安装包，64-Bit Graphical Installer (510 MB)。

清华源下载 Anaconda，打开 https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/。

Anaconda3-5.3.0-MacOSX-x86_64.sh	543.6 MiB	2018-09-28 06:44
Anaconda3-5.3.0-Windows-x86.exe	508.7 MiB	2018-09-28 06:46
Anaconda3-5.3.0-Windows-x86_64.exe	631.4 MiB	2018-09-28 06:46
Anaconda3-5.3.1-Linux-x86.sh	527.3 MiB	2018-11-20 04:00
Anaconda3-5.3.1-Linux-x86_64.sh	637.0 MiB	2018-11-20 04:00
Anaconda3-5.3.1-MacOSX-x86_64.pkg	634.0 MiB	2018-11-20 04:00
Anaconda3-5.3.1-MacOSX-x86_64.sh	543.7 MiB	2018-11-20 04:01
Anaconda3-5.3.1-Windows-x86.exe	509.5 MiB	2018-11-20 04:04
Anaconda3-5.3.1-Windows-x86_64.exe	632.5 MiB	2018-11-20 04:04

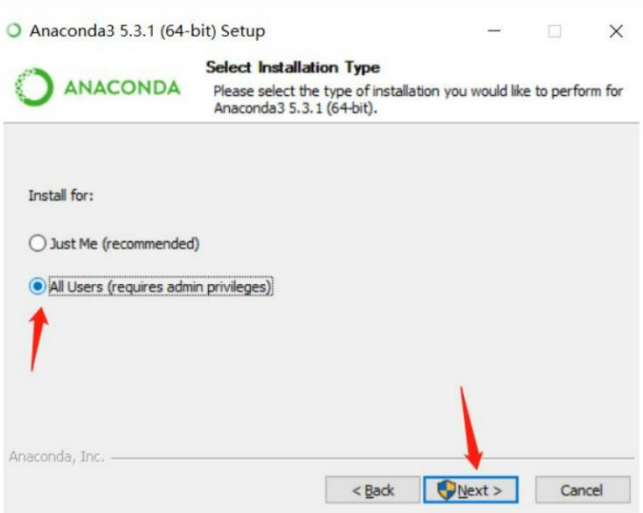
图 1.5 清华源 Anaconda 安装包

如图 1.5 所示，选择对应平台和位数的安装包，例如 Anaconda3-5.3.1-Windows-x86_64.exe。

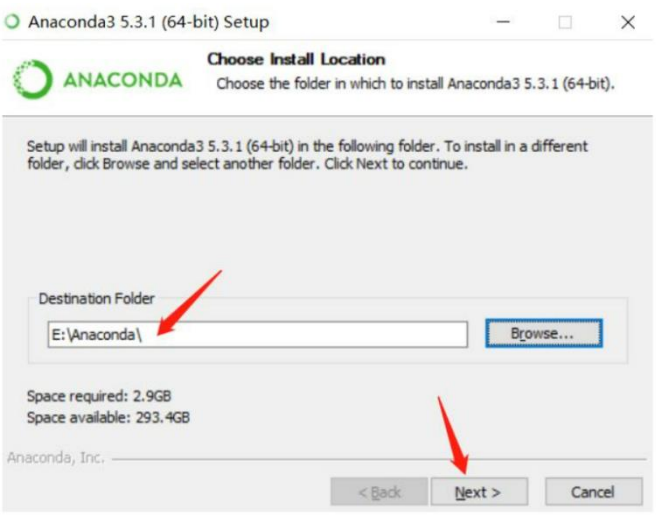
2）安装 Anaconda

双击运行安装包。按照下列图片提示操作。

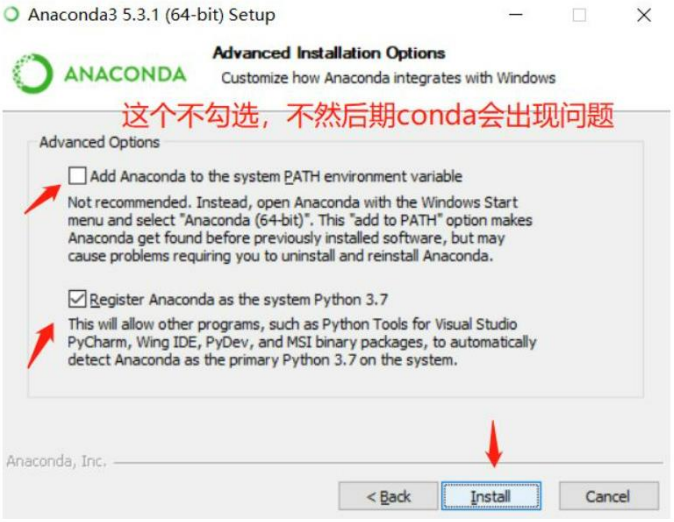
点击 Next，进入下一步。



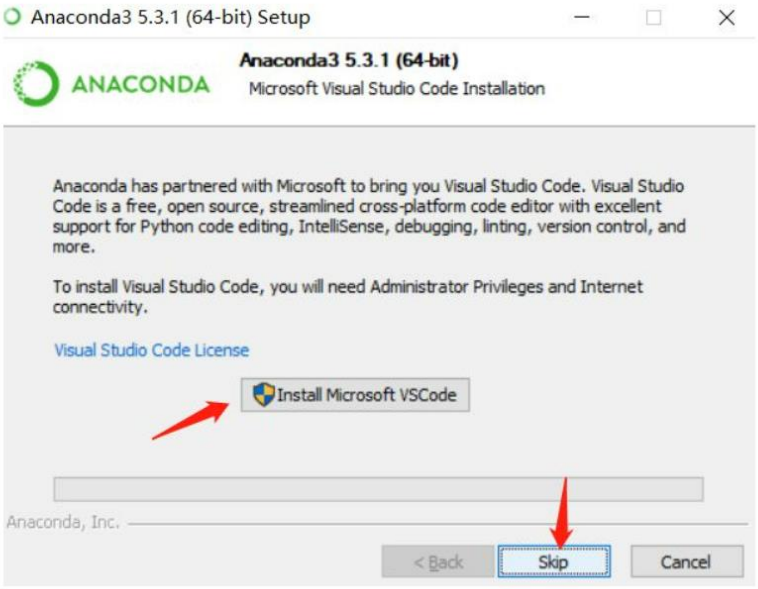
勾选“All Users”，点击进入下一步。



选择安装的文件夹，可以选择非系统盘安装。点击进入下一步。



点击 Install，进入安装界面，按照提示进入下一步。

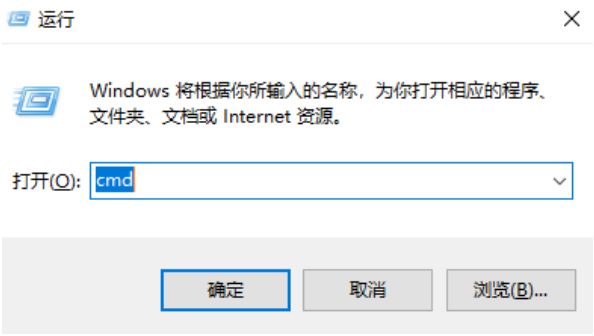


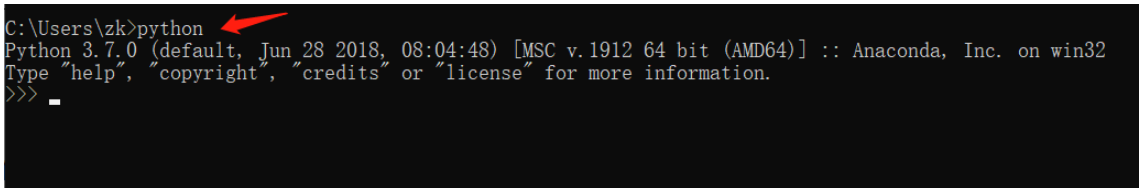
提示安装 VSCode，可以跳过。

点击 Finish，安装完成。

1.2.3 python 环境测试

以 Wondows 环境为例进行 Python 环境测试。快捷键 Win+r 打开 cmd，命令行输入 python，回车。





如果出现上图显示的“>>>”提示符，说明安装成功，Python 的版本为 3.7。

使用 Anaconda 时默认使用 base 环境，可以输入 conda activate base 激活 base 环境，也可以自行创建新的 Python 环境并激活。退出当前环境返回默认 base 环境可以输入 conda deactivate。conda 命令详细教程可以自行百度一下，可以参考一下这个教程 https://zhuanlan.zhihu.com/p/44398592。

2 OpenCV 介绍与安装

2.1 OpenCV 介绍

OpenCV 的全称是 Open Source Computer Vision Library，是一个跨平台的计算机视觉库。OpenCV 是由英特尔公司发起并参与开发，以 BSD 许可证授权发行，可以在商业和研究领域中免费使用。OpenCV 可用于开发实时的图像处理、计算机视觉以及模式识别程序。该程序库也可以使用英特尔公司的 IPP 进行加速处理。OpenCV 用 C++语言编写，它的主要接口也是 C++语言，还有 Python, Java and MATLAB/OCTAVE (版本 2.5)的接口。这些语言的 API 接口函数可以透过在线文档获取。现在也提供对于 C#,Ch,Ruby 的支持。所有新的开发和算法都是用 C++接口。一个使用 CUDA 的 GPU 接口也于 2010 年 9 月开始实现。

2.2 OpenCV 安装

以 Windows 下 Anaconda 的 Python 环境为例介绍 OpenCV 的安装。
使用 pip 安装 opencv-python 包，下载指定版本的包。
例如 pip install opencv-python==3.4.15.55 安装 3.4.15.55 版本的 opencv-python。
使用 conda 安装：conda install opencv-python

```
(py37) C:\Users\Zz>pip install opencv-python==3.4.15.55
Collecting opencv-python==3.4.15.55
  Downloading opencv_python-3.4.15.55-cp37-cp37m-win_amd64.whl (31.1 MB)
     |████████████████████████████████| 31.1 MB 6.8 MB/s
Requirement already satisfied: numpy>=1.14.5 in d:\anaconda3\envs\py37\lib\site-packages (from opencv-python==3.4.15.55)
 (1.21.4)
Installing collected packages: opencv-python
Successfully installed opencv-python-3.4.15.55

(py37) C:\Users\Zz>
```

图 2.1 pip 安装 3.4.15.55 版本的 opencv-python

```
(py37) D:\projects\zhujiao>python
Python 3.7.0 (default, Jun 28 2018, 08:04:48) [MSC v.1912 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>>
```

图 2.2 测试 opencv-python

如上图所示，安装成功。

2.3 OpenCV 简单使用

实例 1：加载图片

```
import cv2

img = cv2.imread('path/to/image/name.jpg', 0)
cv2.imshow("myWindow", img)
cv2.waitKey(0)     # 任意键退出
cv2.destroyAllWindows()   # 关闭窗口
```

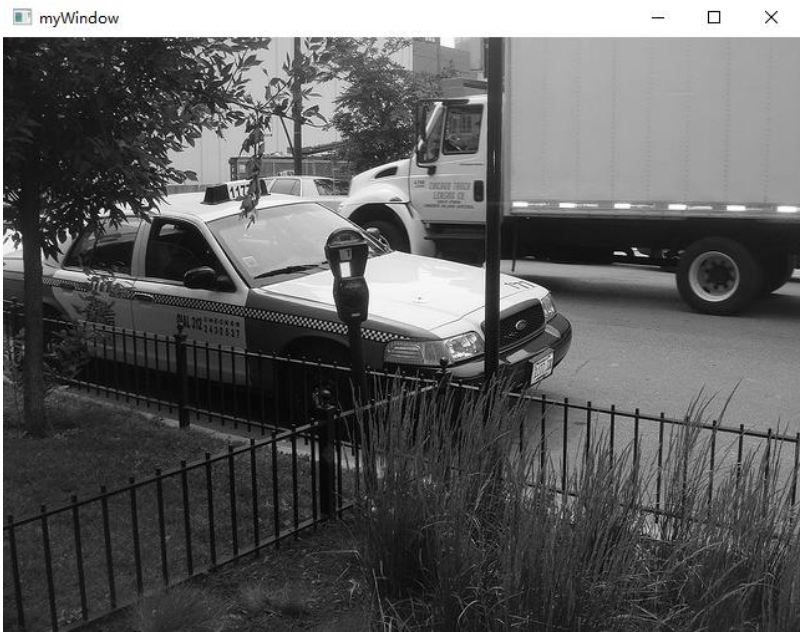


图 2.2 显示图片窗口

其他更加复杂的操作可以参考 https://blog.csdn.net/SGchi/article/details/104463726。

3 SciKit-Image 介绍与安装

3.1 Scikit-Image 介绍

skimage 包的全称是 scikit-image SciKit (toolkit for SciPy) ，它对 scipy.ndimage 进行了扩展，提供了更多的图片处理功能。它是由 python 语言编写的，由 scipy 社区开发和维护。skimage 包由许多的子模块组成，各个子模块提供不同的功能。具体内容可以参考 https://scikit-image.org/。

子模块名称	主要实现功能
io	读取、保存和显示图片或视频
data	提供一些测试图片和样本数据
color	颜色空间变换
filters	图像增强、边缘检测、排序滤波器、自动阈值等
draw	操作于 numpy 数组上的基本图形绘制，包括线条、矩形、圆和文本等
transform	几何变换或其它变换，如旋转、拉伸和拉东变换等
morphology	形态学操作，如开闭运算、骨架提取等
exposure	图片强度调整，如亮度调整、直方图均衡等
feature	特征检测与提取等
measure	图像属性的测量，如相似性或等高线等
segmentation	图像分割
restoration	图像恢复
util	通用函数

3.2 Scikit-Image 安装

使用 pip 安装：pip install scikit-image

使用 conda 安装：conda install scikit-image

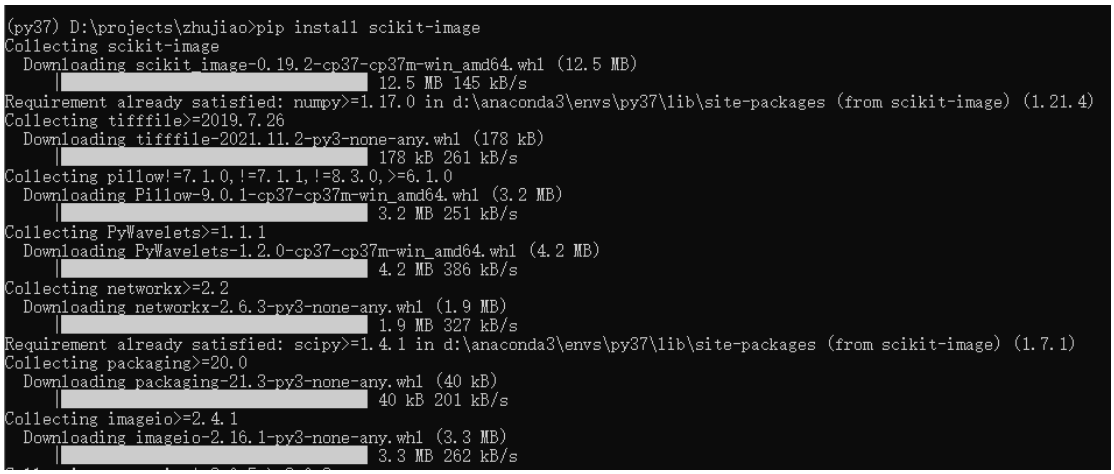


图 3.1 scikit-image 安装

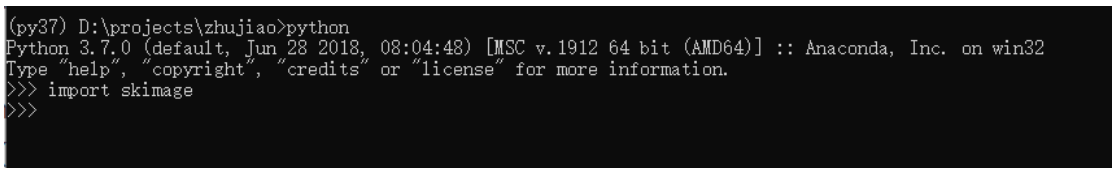


图 3.2 测试 scikit-image

如上图所示，安装 scikit-image 成功。

4 Pillow 介绍与安装

4.1 Pillow 介绍

PIL（Python Imaging Library）提供了图像处理功能。这个库提供了广泛的文件格式支持、高效的内部表示和相当强大的图像处理功能。核心图像库是为快速访问以几种基本像素格式存储的数据而设计的。为通用图像处理工具提供了坚实的基础。
其官网为 https://python-pillow.org/。
学习相关内容可以参考 https://pillow.readthedocs.io/en/stable/

4.2 Pillow 安装

使用 pip 安装：pip install Pillow
使用 conda 安装：conda install Pillow

```
(py37) D:\projects\zhujiao>pip install scikit-image
Collecting scikit-image
  Using cached scikit_image-0.19.2-cp37-cp37m-win_amd64.whl (12.5 MB)
Requirement already satisfied: numpy>=1.17.0 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (1.21.4)
Requirement already satisfied: tifffile>=2019.7.26 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (2021
.11.2)
Requirement already satisfied: packaging>=20.0 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (21.3)
Requirement already satisfied: pillow!=7.1.0,!=7.1.1,!=8.3.0,>=6.1.0 in d:\anaconda3\envs\py37\lib\site-packages (from s
cikit-image) (9.0.1)
Requirement already satisfied: imageio>=2.4.1 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (2.16.1)
Requirement already satisfied: networkx>=2.2 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (2.6.3)
Requirement already satisfied: PyWavelets>=1.1.1 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (1.2.0)

Requirement already satisfied: scipy>=1.4.1 in d:\anaconda3\envs\py37\lib\site-packages (from scikit-image) (1.7.1)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in d:\anaconda3\envs\py37\lib\site-packages (from packaging>=20.
0->scikit-image) (3.0.7)
Installing collected packages: scikit-image
Successfully installed scikit-image-0.19.2
```

图 4.1 安装 Pillow

```
(py37) D:\projects\zhujiao>python
Python 3.7.0 (default, Jun 28 2018, 08:04:48) [MSC v.1912 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import PIL
>>>
```

图 4.2 测试 Pillow

如上图所示，安装 Pillow 成功。

4.3 Pillow 基础教学

实例 1：加载图片

```
from PIL import Image
pil_im = Image.open('name.jpg')  # 读取图片
```

实例 2：转成灰度图

```
from PIL import Image
pil_im = Image.open('empire.jpg').convert('L')  #转成灰度图
```

实例 3：创建缩略图

```
from PIL import Image
pil_im = Image.open('name.jpg')  # 读取图片
pil_im.thumbnail((128,128))
```

实例 4：截取粘贴图片区域

```
from PIL import Image
pil_im = Image.open('name.jpg')  # 读取图片
box = (100,100,400,400)
region = pil_im.crop(box)  # 截取区域

region = region.transpose(Image.ROTATE_180)
pil_im.paste(region,box)  # 粘贴区域
```

实例 5：调整图片尺寸与旋转

from PIL import Image
pil_im = Image.open('name.jpg') # 读取图片
out = pil_im.resize((128,128)) # 改变大小
out = pil_im.rotate(45) # 旋转

5 Numpy 介绍与安装

5.1 Numpy 介绍

NumPy 是 Python 语言的一个扩展程序库。支持高阶大量的维度数组与矩阵运算，此外也针对数组运算提供大量的数学函数库。NumPy 的前身 Numeric 最早是由 Jim Hugunin 与其它协作者共同开发，2005 年，Travis Oliphant 在 Numeric 中结合了另一个同性质的程序库 Numarray 的特色，并加入了其它扩展而开发了 NumPy。NumPy 为开放源代码并且由许多协作者共同维护开发。在图像处理中需要处理大量的矩阵数据，Numpy 在处理矩阵方面比较舒服。可在其官网上 https://numpy.org/获取更多信息。

5.2 Numpy 安装

使用 pip 安装：pip install numpy
使用 conda 安装：conda install numpy

```
(py37) D:\projects\zhujiao>pip install numpy
Collecting numpy
  Downloading numpy-1.21.5-cp37-cp37m-win_amd64.whl (14.0 MB)
     |████████████████████████████████| 14.0 MB 37 kB/s
Installing collected packages: numpy
```

图 5.1 安装 Numpy

```
(py37) D:\projects\zhujiao>python
Python 3.7.0 (default, Jun 28 2018, 08:04:48) [MSC v.1912 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>>
```

图 5.2 测试 Numpy

如上图所示，安装成功。

5.3 Numpy 基础教学

实例 1：图像数组表示

当载入图像时，我们通过调用 array() 方法将图像转换成 NumPy 的数组对象，但当时并没有进行详细介绍。NumPy 中的数组对象是多维的，可以用来表示向量、矩阵和图像。一个数组对象很像一个列表（或者是列表的列表），但是数组中所有的元素必须具有相同的数据类型。除非创建数组对象时指定数据类型，否则数据类型会按照数据的类型自动确定。

对于图像数据，下面的例子阐述了这一点：

```
im = array(Image.open('empire.jpg'))
print im.shape, im.dtype

im = array(Image.open('empire.jpg').convert('L'),'f')
print im.shape, im.dtype
```

控制台输出结果如下所示：

```
(800, 569, 3) uint8
(800, 569) float32
```

每行的第一个元组表示图像数组的大小（行、列、颜色通道），紧接着的字符串表示数组元素的数据类型。因为图像通常被编码成无符号八位整数（uint8），所以在第一种情况下，载入图像并将其转换到数组中，数组的数据类型为“uint8”。在第二种情况下，对图像进行灰度化处理，并且在创建数组时使用额外的参数“f”；该参数将数据类型转换为浮点型。注意，由于灰度图像没有颜色信息，所以在形状元组中，它只有两个数值。

数组中的元素可以使用下标访问。位于坐标 i、j，以及颜色通道 k 的像素值可以像下面这样访问：

```
value = im[i,j,k]
```

多个数组元素可以使用数组切片方式访问。切片方式返回的是以指定间隔下标访问该数组的元素值。下面是有关灰度图像的一些例子：

```
im[i,:] = im[j,:]        # 将第 j 行的数值赋值给第 i 行
im[:,i] = 100            # 将第 i 列的所有数值设为 100
im[:100,:50].sum()       # 计算前 100 行、前 50 列所有数值的和
im[50:100,50:100]        # 50~100 行，50~100 列（不包括第 100 行和第 100 列）
im[i].mean()             # 第 i 行所有数值的平均值
im[:,-1]                 # 最后一列
im[-2,:] (or im[-2])     # 倒数第二行
```

注意，示例仅仅使用一个下标访问数组。如果仅使用一个下标，则该下标为行下标。注意，在最后几个例子中，负数切片表示从最后一个元素逆向计数。我们将会频繁地使用切片技术访问像素值，这也是一个很重要的思想。

实例 2：灰度变换

将图像读入 NumPy 数组对象后，我们可以对它们执行任意数学操作。一个简单的例子就是图像的灰度变换。考虑任意函数 f，它将 0...255 区间（或者 0...1 区间）映射到自身（意思是说，输出区间的范围和输入区间的范围相同）。下面是关于灰度变换的一些

例子：

```
from PIL import Image
from numpy import *
im = array(Image.open('empire.jpg').convert('L'))
im2 = 255 - im # 对图像进行反相处理
im3 = (100.0/255) * im + 100 # 将图像像素值变换到 100...200 区间
im4 = 255.0 * (im/255.0)**2 # 对图像像素值求平方后得到的图像
```

第一个例子将灰度图像进行反相处理；第二个例子将图像的像素值变换到 100...200 区间；第三个例子对图像使用二次函数变换，使较暗的像素值变得更小。图 1-4 为所使用的变换函数图像。图 1-5 是输出的图像结果。你可以使用下面的命令查看图像中的最小和最大像素值：

```
print int(im.min()), int(im.max())
```

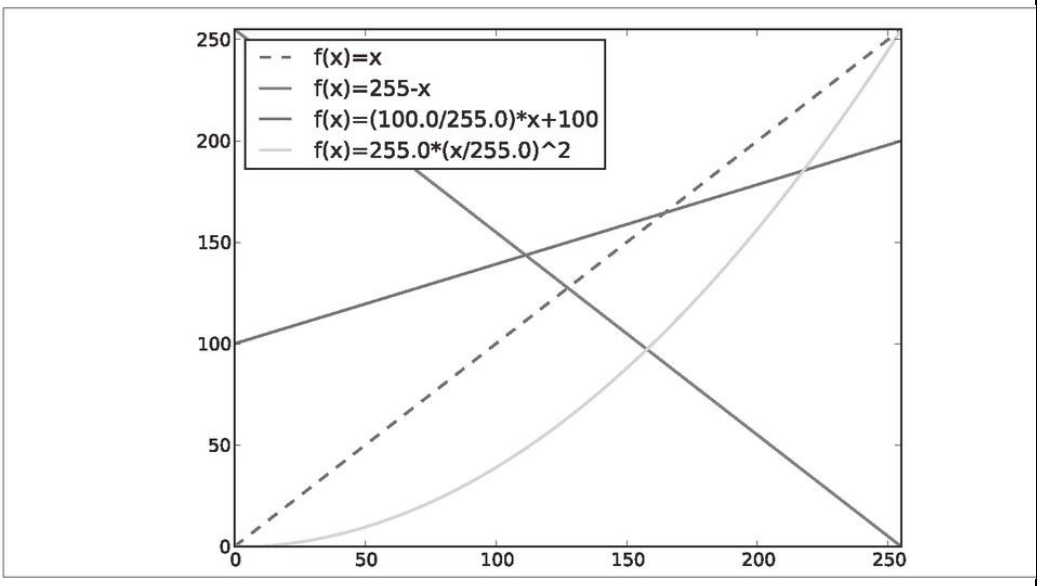


图 1-4：灰度变换示例。三个例子中所使用函数的图像，其中虚线表示恒等变换

图 1-5：灰度变换。

对图像应用图 1-4 中的函数：f(x)=255-x 对图像进行反相处理（左）；f(x)=(100/255)x+100 对图像进行变换（中）；f(x)=255(x/255)2 对图像做二次变换（右）

如果试着对上面例子查看最小值和最大值，可以得到下面的输出结果：

```
2 255
0 253
```

100 200

0 255

array() 变换的相反操作可以使用 PIL 的 fromarray() 函数完成：

pil_im = Image.fromarray(im)

如果你通过一些操作将“uint8”数据类型转换为其他数据类型，比如之前例子中的 im3 或者 im4，那么在创建 PIL 图像之前，需要将数据类型转换回来：

pil_im = Image.fromarray(uint8(im))

如果你并不十分确定输入数据的类型，安全起见，应该先转换回来。注意，NumPy 总是将数组数据类型转换成能够表示数据的“最低”数据类型。对浮点数做乘积或除法操作会使整数类型的数组变成浮点类型。

实例 3：图像缩放

NumPy 的数组对象是我们处理图像和数据的主要工具。想要对图像进行缩放处理没有现成简单的方法。我们可以使用之前 PIL 对图像对象转换的操作，写一个简单的用于图像缩放的函数。把下面的函数添加到 imtool.py 文件里：

```
def imresize(im,sz):
    """ 使用 PIL 对象重新定义图像数组的大小"""
    pil_im = Image.fromarray(uint8(im))
    return array(pil_im.resize(sz))
```

6 Matplotlib 介绍与安装

6.1 Matplotlib 介绍

matplotlib 是 Python 语言及其数值计算库 NumPy 的绘图库。它提供了一个面向对象的 API，用于使用通用 GUI 工具包（如 Tkinter、wxPython、Qt 或 GTK）将绘图嵌入到应用程序中。它还有一个基于状态机（如 OpenGL）的过程式编程“pylab”接口，其设计与 MATLAB 非常类似，但不推荐使用。SciPy 使用 matplotlib 进行图形绘制。matplotlib 最初由 John D. Hunter 撰写，从此它拥有一个活跃的开发社区，并根据 BSD 许可证发布。 在 John D. Hunter 于 2012 年 8 月去世前不久，Michael Droettboom 被提名为 matplotlib 的主要开发者，Thomas Caswell 也加入了他的行列。Matplotlib 2.0.x 支持 Python 2.7 到 3.10 版本。Matplotlib 1.2 是第一个支持 Python 3.x 的版本。Matplotlib 1.4 是支持 Python 2.6 的最后一个版本。Matplotlib 已签署 Python 3 声明，承诺在 2020 年后不再支持 Python 2。可以在学习网站上 http://www.matplotlib.org.cn/intro/获取更多内容。

6.2 Matplotlib 安装

使用 pip 安装：pip install matplotlib
使用 conda 安装：conda install matplotlib

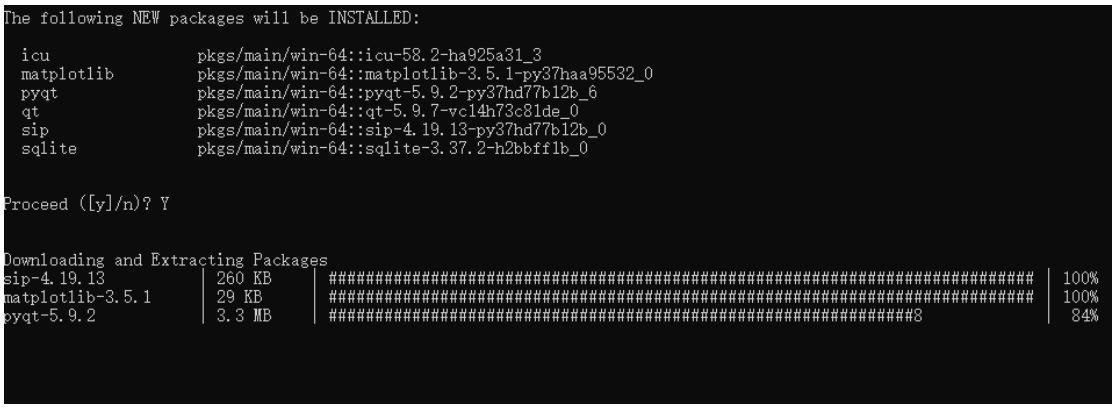


图 6.1 安装 matplotlib

图 6.2 测试 matplotlib

如图 6.2，安装成功。

6.3 Matplotlib 基础教学

尽管 Matplotlib 可以绘制出较好的条形图、饼状图、散点图等，但是对于大多数计算机视觉应用来说，仅仅需要用到几个绘图命令。最重要的是，我们想用点和线来表示一些事物，比如兴趣点、对应点以及检测出的物体。

实例 1：绘制简单图像

```
from PIL import Image
from pylab import *

# 读取图像到数组中
im = array(Image.open('empire.jpg'))

# 绘制图像
imshow(im)
```

```
# 一些点
x = [100,100,400,400]
y = [200,500,200,500]

# 使用红色星状标记绘制点
plot(x,y,'r*')

# 绘制连接前两个点的线
plot(x[:2],y[:2])

# 添加标题，显示绘制的图像
title('Plotting: "empire.jpg"')

axis('off')  # 不显示坐标轴

show()
```

上面的代码首先绘制出原始图像，然后在 x 和 y 列表中给定点的 *x* 坐标和 *y* 坐标上绘制出红色星状标记点，最后在两个列表表示的前两个点之间绘制一条线段（默认为蓝色）。

在绘图时，有很多选项可以控制图像的颜色和样式。最有用的一些短命令如表 6-1、表 6-2 和表 6-3 所示。使用方法见下面的例子：

```
plot(x,y)           # 默认为蓝色实线
plot(x,y,'r*')      # 红色星状标记
plot(x,y,'go-')     # 带有圆圈标记的绿线
plot(x,y,'ks:')     # 带有正方形标记的黑色虚线
```

表 6-1：用 PyLab 库绘图的基本颜色格式命令

颜色	
'b'	蓝色
'g'	绿色
'r'	红色
'c'	青色
'm'	品红
'y'	黄色
'k'	黑色

颜色	
'w'	白色

表 6-2：用 PyLab 库绘图的基本线型格式命令

线型	
'-'	实线
'--'	虚线
':'	点线

表 6-3：用 PyLab 库绘图的基本绘制标记格式命

标记	
'.'	点
'o'	圆圈
's'	正方形
'*'	星形
'+'	加号
'x'	叉号

实例 2：绘制图像轮廓和直方图

下面来看两个特别的绘图示例：图像的轮廓和直方图。绘制图像的轮廓（或者其他二维函数的等轮廓线）在工作中非常有用。因为绘制轮廓需要对每个坐标 [x, y] 的像素值施加同一个阈值，所以首先需要将图像灰度化：

```
from PIL import Image
from pylab import *

# 读取图像到数组中
im = array(Image.open('empire.jpg').convert('L'))

# 新建一个图像
figure()
```

```
# 不使用颜色信息
gray()
# 在原点的左上角显示轮廓图像
contour(im, origin='image')
axis('equal')
axis('off')
```

这里用 PIL 的 convert() 方法将图像转换成灰度图像。图像的直方图用来表征该图像像素值的分布情况。用一定数目的小区间（bin）来指定表征像素值的范围，每个小区间会得到落入该小区间表示范围的像素数目。该（灰度）图像的直方图可以使用 hist() 函数绘制：

```
figure()
hist(im.flatten(),128)
show()
```

hist() 函数的第二个参数指定小区间的数目。需要注意的是，因为 hist() 只接受一维数组作为输入，所以我们在绘制图像直方图之前，必须先对图像进行压平处理。flatten() 方法将任意数组按照行优先准则转换成一维数组。图 6.3 为等轮廓线和直方图图像。

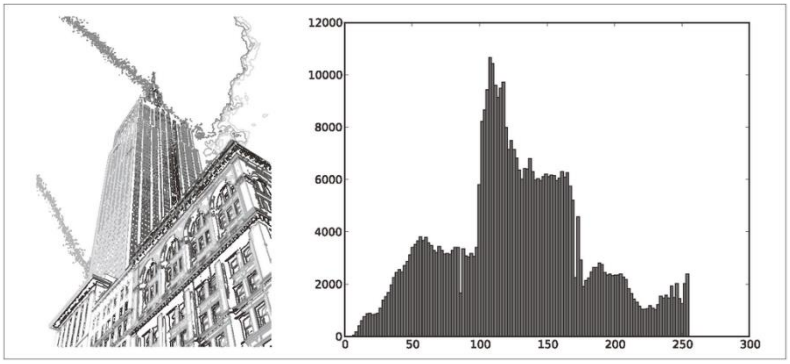


图 6.3：用 Matplotlib 绘制图像等轮廓线和直方图

实例 3：交互式标注

有时用户需要和某些应用交互，例如在一幅图像中标记一些点，或者标注一些训练数据。PyLab 库中的 ginput() 函数就可以实现交互式标注。下面是一个简短的例子：

```
from PIL import Image
from pylab import *

im = array(Image.open('empire.jpg'))
imshow(im)
print 'Please click 3 points'
x = ginput(3)
print 'you clicked:',x
```

```
show()
```

上面的脚本首先绘制一幅图像，然后等待用户在绘图窗口的图像区域点击三次。程序将这些点击的坐标 [x, y] 自动保存在 x 列表里。