

实验一 Python 数字图像处理初步

一、实验目的与要求

掌握 OpenCV-python 安装；掌握 python 语言与 openCV 的交互方法；掌握对图像的基本操作，如：读取，显示，保存，以及对视频的读取，保存；掌握使用 OpenCV 进行简单运作，如：画图，对鼠标的事理事件等。

二、实验原理及知识点

(1) OpenCV-python 安装：

A. 通过 Anaconda 安装 python 环境。

Anaconda 指的是一个开源的 Python 发行版本，其包含了 conda、Python 等 180 多个科学包及其依赖项。通过 Anaconda 安装 python 包可以简化很多安装配置的问题。也可以通过其虚拟环境（注：不是虚拟机）在同一操作系统上使用多个不同 python 版本或配置。

下载 Anaconda:

<https://www.anaconda.com/download/>

可以选择 Python2.7 版本或 Python3.7 版本；两个都安装也没有问题。我们会通过不同虚拟环境使用相应的 python 版本。

进入 Anaconda 的安装目录的 Scripts 文件夹运行 activate 命令激活虚拟环境。例如：

```
C:\Users\zdxue\Anaconda2\Scripts>activate
```

可以在命令行键入 python 并执行。运行结果如下：

```
(root) C:\Users\zdxue\Anaconda2\Scripts>python
Python 2.7.14 [Anaconda, Inc.] (default, Oct 15 2017, 03:34:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

可以发现当前使用的 python 版本为 2.7.14.

若安装的 Anacoda3.7, 激活虚拟环境的目录或 python 提示应该略有不同。

B. 激活虚拟环境后，通过 pip 命令安装 numpy 命令如下：

pip install numpy 检查 numpy 是否已经安装 (提示 : anaconda 或自动安装 numpy , 也可以通过 pip 安装一些 Anaconda 没有默认安装的软件包) 。

输入 python 执行后 , import numpy 包 , 检查 numpy 是否安装成功 , 若无错误显示 , 则表明成功 , 如下图。

```
(root) C:\Users\zdxue\Anaconda2\Scripts>python
Python 2.7.14 |Anaconda, Inc.| (default, Oct 15 2017, 03:34:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>>
```

可以键入 exit() 退出 python 交互环境.

C . 安装 OpenCV-python

I . 配置系统环境变量 Path , 增加 <安装目录>/Anaconda2/Scripts 目录。

II . 打开 cmd 进入自己的工作目录 : 例如 : D:\OpenCVcourse

III . 键入如下命令创建自己的以 OpenCV 命名的 python 虚拟环境。

```
conda create -n OpenCV
```

IV . 激活 OpenCV 虚拟环境 , 命令如下。

```
Activate OpenCV
```

输出效果图 :

```
D:\opencvcourse>activate opencv
(opencv) D:\opencvcourse>
```

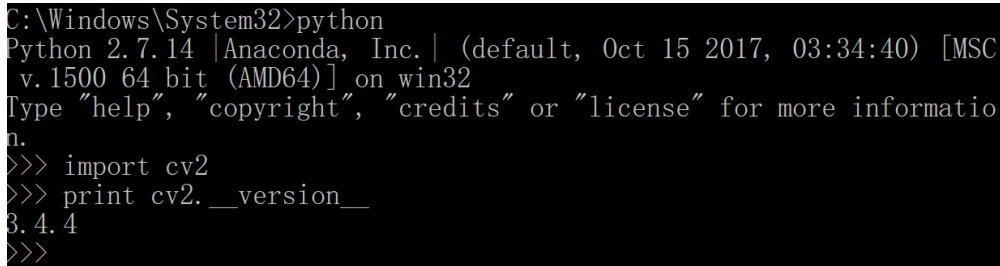
V . 通过 pip 命令安装 python-OpenCV :

pip install OpenCV-python , 然后打开键入 python 回车进入 python 交互环境。

继续输入：

```
>>> import cv2
>>> print cv2.__version__    #For python2
Print(cv2.__version__)    # For python3
```

若没有提示错误，表示安装成功。效果如下：



```
C:\Windows\System32>python
Python 2.7.14 |Anaconda, Inc.| (default, Oct 15 2017, 03:34:40) [MSC
v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more informatio
n.
>>> import cv2
>>> print cv2.__version__
3.4.4
>>>
```

(另外一种安装 Opencv 的方法：

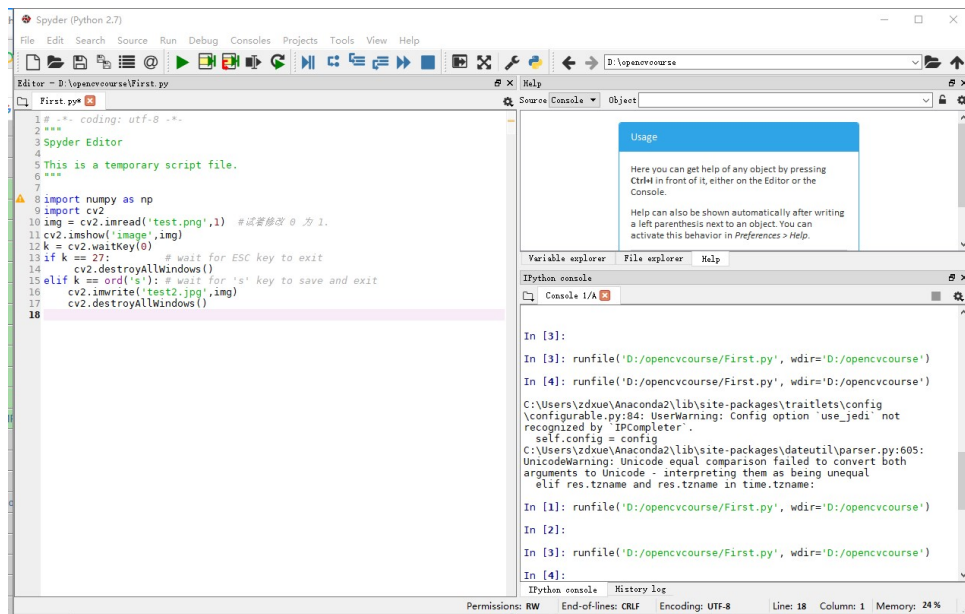
```
conda install -c https://conda.anaconda.org/menpo opencv)
```

(2)图像读取与保存

下面给出读取、显示，保存一幅图像的
程序。

```
import numpy as np
import cv2
img = cv2.imread('test.png',0)  #试着修改 0 为 1.
cv2.imshow('image',img)
k = cv2.waitKey(0)
if k == 27:          # wait for ESC key to exit
    cv2.destroyAllWindows()
elif k == ord('s'): # wait for 's' key to save and exit
    cv2.imwrite('test2.jpg',img)
    cv2.destroyAllWindows()
```

键入 exit()退出 python 交互环境，打开 Anaconda 自带的 python GUI IDE 程序 spyder。键入 spyder 回车。将上述代码拷贝到编辑区，并进行编辑使其符合 Python 语法规则，保存到 d:\OpenCVcourse 目录。并将本文档附图另存为 test.png 到同一目录。点击绿色三角形运行程序。如下图：



(Spyder 无法启动的解决方案：

可能是 pyqt 未安装，或版本不是最新，请更新至最新即可。

方法：终端输入 `conda install pyqt==5.6` 或者更新版本。)

(3) 读取视频文件，显示视频，保存视频文件
读取与显示

用到 `cv2.VideoCapture()`和 `cv2.VideoWrite()`函数。在 `cv2.VideoCapture` 函

数指定输入视频文件名称，在播放每一帧时，使用 `cv2.waitKey()` 设置适当的持续时间。如果设置的太低视频就会播放的非常快，如果设置的太高就会播放的很慢（你可以使用这种方法控制视频的播放速度）。通常情况下 25 毫秒就可以了。视频读取与显示方法如下：

```
import numpy as np
import cv2
cap = cv2.VideoCapture('vtest.avi')
while(cap.isOpened()):
    ret, frame = cap.read()
    if ret:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        cv2.imshow('frame',gray)
        if cv2.waitKey(1) & 0xFF == ord('q') :
            break
cap.release()
```

```
cv2.destroyAllWindows()
```

运行结果：



◆ 从摄像头获取图像并保存为视频文件。

在捕获视频并对每一帧都进行加工之后，往往需要保存处理后的视频。对于图片来说，只需要使用 `cv2.imwrite()` 进行保存。但对于视频来说就要多做点工作。需要创建一个 `VideoWriter` 的对象和确定一个输出文件的名字。接下来指定 **FourCC 编码**、播放频率、帧尺寸和 `isColor` 标签。如果 `isColor` 是 `True`，用彩色存储，否则就用灰度。

`FourCC` 就是一个 4 字节码，用来确定视频的编码格式。可用的编码列表可以从 fourcc.org 查到。`FourCC` 是平台依赖的。

下面的代码是从摄像头中捕获视频，沿水平方向旋转每一帧并保存它。

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0) #摄像头编号。

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')# 注意编码器
out = cv2.VideoWriter('output.avi',fourcc, 20.0, (640,480))

while(cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)
        # write the flipped frame
        out.write(frame)
        cv2.imshow('frame',frame)
```

```

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    # Release everything if job is finished
    cap.release()
    out.release()
    cv2.destroyAllWindows()

```

(3) OpenCV 画图

- 学习使用 OpenCV 绘制不同几何图形
- 学习这些函数：cv2.line(), cv2.circle(), cv2.rectangle(), cv2.ellipse(), cv2.putText() 等。

上面所有的这些绘图函数需要设置下面这些参数：

- **img**：想要绘制图形的那幅图像。
- **color**：形状的颜色。以 RGB 为例，需要传入一个元组，例如：(255,0,0) 代表蓝色（OPENCV 表示彩色三元组顺序是 BGR，不是 RGB，）。对于灰度图只需要传入灰度值。
- **thickness**：线条的粗细。如果给一个闭合图形设置为 -1，那么这个图形就会被填充。默认值是 1。
- **linetype**：线条的类型，8 连接，抗锯齿等。默认情况是 8 连接。cv2.LINE_AA 为抗锯齿，这样看起来会非常平滑。

◆ 画线

要画一条线，需要告诉函数这条线的起点和终点。下面的代码会画一条从左上到右下角的蓝色线段。

```

import numpy as np
import cv2

```

```

# Create a black image

```

```

img=np.zeros((512,512,3), np.uint8) # np.uint8 是数据类型

```

```

# Draw a diagonal blue line with thickness of 5 px

```

```

cv2.line(img,(0,0),(511,511),(255,0,0),5)

```

◆ 画矩形

要画一个矩形，需要告诉函数的左上角顶点和右下角顶点的坐标。这次会在图像的右上角画一个绿色的矩形。

```

cv2.rectangle(img,(384,0),(510,128),(0,255,0),3)

```

◆ 画圆

要画圆的话，只需要指定圆形的中心点坐标和半径大小。我们在上面的矩形中画一个圆。

```

cv2.circle(img,(447,63), 63, (0,0,255), -1)

```

◆ 画椭圆

画椭圆比较复杂，要多输入几个参数。一个参数是中心点的位置坐标。下一个参数是长轴和短轴的长度。椭圆沿逆时针方向旋转的角度。椭圆会沿着顺时针方向起始的角度和结束角度，如果是 0 和 360，就是整个椭圆。查看 cv2.ellipse()

可以得到更多信息。下面的例子是在图片的中心绘制半个椭圆。

```
cv2.ellipse(img,(256,256),(100,50),0,0,180,255,-1)
```

◆ 画多边形

画多边形，需要指点每个顶点的坐标。用这些点的坐标构建一个大小等于行数 X1X2 的数组，行数就是点的数目。这个数组的数据类型必须为 `np.int32`。这里画一个黄色的具有四个顶点的多边形。

```
pts=np.array([[10,5],[20,30],[70,20],[50,10]], np.int32)
pts=pts.reshape((-1,1,2))
cv2.polylines(img,[pts],True,(0,255,255))
```

这里 `reshape` 的第一个参数为-1，表明这一维的长度是根据后面的维度的计算出来的。

(4) 绘制文字

OpenCV 处要在图片上绘制文字，需要设置下列参数：

- 要绘制的文字
- 要绘制的位置
- 字体类型（通过查看 `cv2.putText()` 的文档找到支持的字体）
- 字体的大小
- 文字的一般属性如颜色，粗细，线条的类型等。为了更好看一点推荐使用

```
linetype=cv2.LINE_AA。
```

在图像上绘制白色的 OpenCV。

```
font=cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(img, 'OpenCV', (10, 500), font,
            4, (255, 255, 255), 2)
```

(6) 理鼠标事件

.学习使用 OpenCV 处理鼠标事件

.将要学习的函数是：`cv2.setMouseCallback()`

现在我们来创建一个更好的程序。这次要完成的任务是根据选择的模式在拖动鼠标时绘制矩形或者是圆圈（就像画图程序中一样）。所以回调函数包含两部分，一部分画矩形，一部分画圆圈。这是一个典型的例他可以帮助我们更好理解与构建人机交互式程序，比如物体跟踪，图像分割等。

示例：

```
import cv2
import numpy as np
# 当鼠标按下时变为 True
drawing=False
# 如果 mode 为 true 绘制矩形。按下'm' 变成绘制曲线。
```

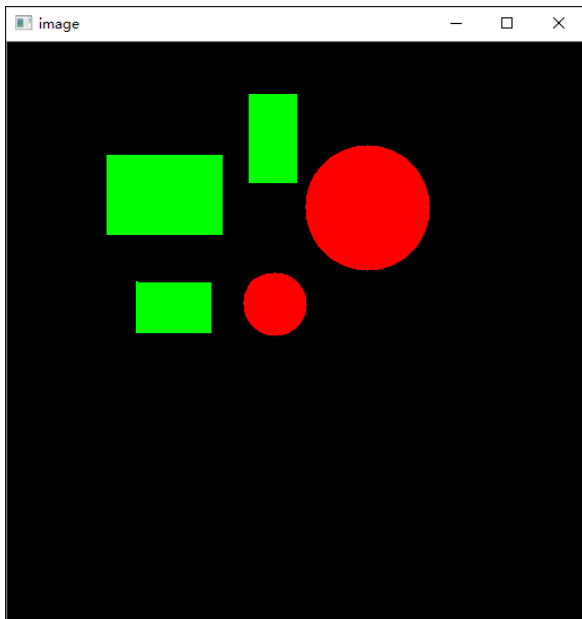
```

mode=True
ix,iy=-1,-1
# 创建回调函数
def draw_circle(event,x,y,flags,param):
    global ix,iy,drawing,mode
    # 当按下左键是返回起始位置坐标
    if event==cv2.EVENT_LBUTTONDOWN:
        drawing=True
        ix,iy=x,y
        # 当鼠标左键按下并移动是绘制图形。event 可以查看移动, flag 查看是否按下
    elif event==cv2.EVENT_MOUSEMOVE and
flags==cv2.EVENT_FLAG_LBUTTON:
        if drawing==True:
            if mode==True:
                cv2.rectangle(img,(ix,iy),(x,y),(0,255,0),-1)
            else:
                r=int(np.sqrt((x-ix)**2+(y-iy)**2))
                cv2.circle(img,(x,y),r,(0,0,255),-1)
        # 当鼠标松开停止绘画。
    elif event==cv2.EVENT_LBUTTONUP:
        if mode==True:
            cv2.rectangle(img,(ix,iy),(x,y),(0,255,0),-1)
        else:
            cv2.circle(img,(x,y),5,(0,0,255),-1)

img=np.zeros((512,512,3),np.uint8)
cv2.namedWindow('image')
cv2.setMouseCallback('image',draw_circle)
while(1):
    cv2.imshow('image',img)
    k=cv2.waitKey(1)&0xFF
    if k==ord('m'):
        mode=not mode
    elif k==27:
        break
cv2.destroyAllWindows()

```

运行后可以，操作鼠标绘制矩形、圆形。实例结果如下：



这个例子简单的描述了鼠标的交互事件。

思考:如何修改矩形的颜色?

三、实验内容及步骤

1. 利用 `imread()` 函数读取一幅图像;利用 `imshow()` 函数来显示这幅图像; 利用 `imwrite()` 函数来保存这幅图像;
2. 读写视频
3. OpenCV 画图
4. OpenCV 处理鼠标事件
5. 请参考本实验手册, 实现一个在视频叠加字幕的程序。鼠标点击屏幕, 关闭或打开字幕。

四、考核要点

- 1、熟悉在 Python 中如何读入图像、如何获取图像文件的相关信息、如何显示图像及保存图像等, 熟悉相关的处理函数。

五、实验仪器与软件

- (1) PC 计算机
- (2) Python 语言开发环境, numpy 安装包
- (3) 实验所需要的图片

六、实验报告要求

描述实验的基本步骤, 用数据和图片给出各个步骤中取得的实验结果和源代码, 并进行必要的讨论, 必须包括原始图像及其计算/处理后的图像。

七、实验图像



test.jpg



mogu.jpg