



周围的人都比你厉害，你才会慢慢变强

公告

昵称： 山上有风景  
园龄： 1年11个月  
粉丝： 170  
关注： 19  
+加关注

<	2019年12月											>
日	一	二	三	四	五	六						
1	2	3	4	5	6	7						
8	9	10	11	12	13	14						
15	16	17	18	19	20	21						
22	23	24	25	26	27	28						
29	30	31	1	2	3	4						
5	6	7	8	9	10	11						

搜索

找找看

常用链接

我的随笔  
我的评论  
我的参与  
最新评论  
我的标签

我的标签

STL(18)  
SDN(15)  
ThinkPHP3.2(1)

积分与排名

积分 - 211835  
排名 - 1998

随笔分类

OpenCV---Numpy数组的使用以及创建图片

目录

- 一：对头像的所有像素进行访问，并UI图像进行像素取反
- (一) for循环取反
- (二) 使用内置方法取反（直接使用c代码执行，效率更高）
- 二：使用Numpy数组，创建图片
- (一) 使用多个信道创建图片
- (二) 使用单个信道创建图像（灰度图像）
- 三：补充Numpy的使用
- (一) 二维数组的使用（选择正确的类型）
- (二) 维数转换reshape
- (三) 使用array自定义数组

一：对头像的所有像素进行访问，并UI图像进行像素取反

(一) for循环取反

```
import cv2 as cv
import numpy as np

def access_pixels(image):    #对图像的所有像素进行访问
    print(image.size)
    height,width,channel = image.shape    #每个像素3个通道，通道顺序b,g,r
    print("height:%s\r\nwidth:%s\r\nchannel:%s\r\n"%(height,width,channel))
    ...

    height:608
    width:343
    channel:3
    ...

    for row in range(height):
        for col in range(width):
            for c in range(channel):    #循环会变慢,经过625632循环
```

00

关注 | 顶部 | 评论

- C/C++(74)
- Html(2)
- Java(33)
- Javascript(19)
- OpenCV(29)
- PHP(2)
- Python(155)
- STL泛型编程(18)
- 单片机笔记（复习用）(3)
- 计算机网络(32)
- 其他知识(14)
- 设计模式(27)
- 数据结构(57)
- 数据库(8)
- 算法习题(43)
- 算法训练营
- 随笔所想(4)
- 图形界面编程
- 正则表达式(2)
- 转载推文(4)

随笔档案

- 2019年11月(5)
- 2019年10月(32)
- 2019年9月(21)
- 2019年7月(10)
- 2019年5月(8)
- 2019年4月(25)
- 2019年3月(8)
- 2019年2月(1)
- 2019年1月(12)
- 2018年12月(19)
- 2018年9月(5)
- 2018年8月(95)
- 2018年7月(78)
- 2018年6月(26)
- 2018年5月(17)
- 2018年4月(22)
- 2018年3月(111)

最新评论

- 1. Re:python---websocket 的使用  
网上的方法都不行，换成gbk会报如下的错：  
IndexError: index out of range  
--缘分天空0320
- 2. Re:python---websocket 的使用

```
pv = image[row,col,c]
image[row,col,c] = 255 - pv #像素取反
cv.imshow("pixels_demo",image)

src = cv.imread("./1.png") #读取图片
cv.namedWindow("input image",cv.WINDOW_AUTOSIZE) #创建GUI窗口,形式为自适应
cv.imshow("input image",src) #通过名字将图像和窗口联系
t1 = cv.getTickCount() #获取时间，用于精度计时，操作系统启动所经过（elapsed）的毫秒数
access_pixels(src)
t2 = cv.getTickCount()
print((t2-t1)/cv.getTickFrequency()) #getTickFrequency()是获取一秒钟结果的点数，获取秒数
cv.waitKey(0) #等待用户操作，里面等待参数是毫秒，我们填写0，代表是永远，等待用户操作
cv.destroyAllWindows() #销毁所有窗口
```

  
625632  
height:608  
width:343  
channel:3  
  
15.740029368334588 #经历了15秒，是十分耗时的循环，我们可以使用Numpy数组访问，更加方便快捷  


00

关注 | 顶部 | 评论

@ 缘分天空0320太久没用，忘了。应该是字符编码问题吧。这类问题网上应该可以很容易找到方法解决。你看看Python的默认编码和代码是不是一致。一般就是gbk和utf8之间出错...

--山上有风景

3. Re:python---websocket的使用

您好，用了您的代码，报如下错误，麻烦问下如何解决呢？

Traceback (most recent call last): File "server3.py", line 101, in <module...

--缘分天空0320

4. Re:数据结构（六）查找---线性索引查找

请问最后倒序排序的那个代码怎么实现的？可以发一下吗？

--Viki-

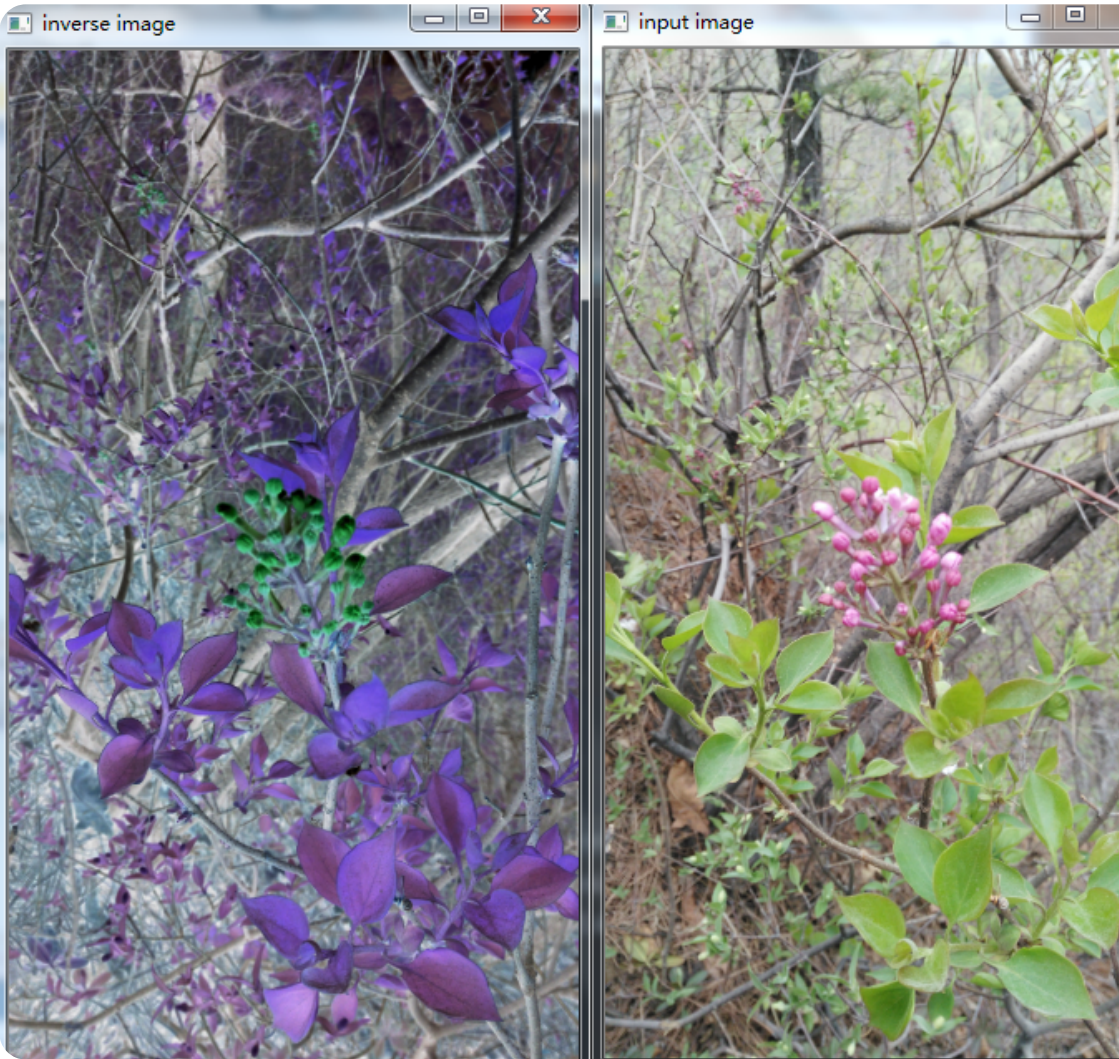
5. Re:SDN实验---Ryu的源码分析

@ 山上有风景谢谢！ ...

--iRoy\_33

- 阅读排行榜
- python---websocket的使用(17249)
  - OpenCV---图像二值化(12510)
  - OpenCV---模板匹配matchTemplate(11531)
  - OpenCV---直线检测(8206)
  - python---基础知识回顾（九）图形用户界面-----wxPython(7986)

- 评论排行榜
- python---基础知识回顾（九）图形用户界面-----Tkinter(4)
  - python---websocket的使用(3)
  - SDN实验---Ryu的源码分析(3)
  - 数据结构（三）串---KMP模式匹配算法之获取next数组(2)



## （二）使用内置方法取反（直接使用c代码执行，效率更高）

```
def inverse(image):  
    img = cv.bitwise_not(image)  
    cv.imshow("inverse image",img)  
  
t1 = cv.getTickCount() #获取时间，用于精度计时，操作系统启动所经过（elapsed）的毫秒数  
inverse(src)  
t2 = cv.getTickCount()  
  
0.09940230583146789
```

## 二：使用Numpy数组，创建图片

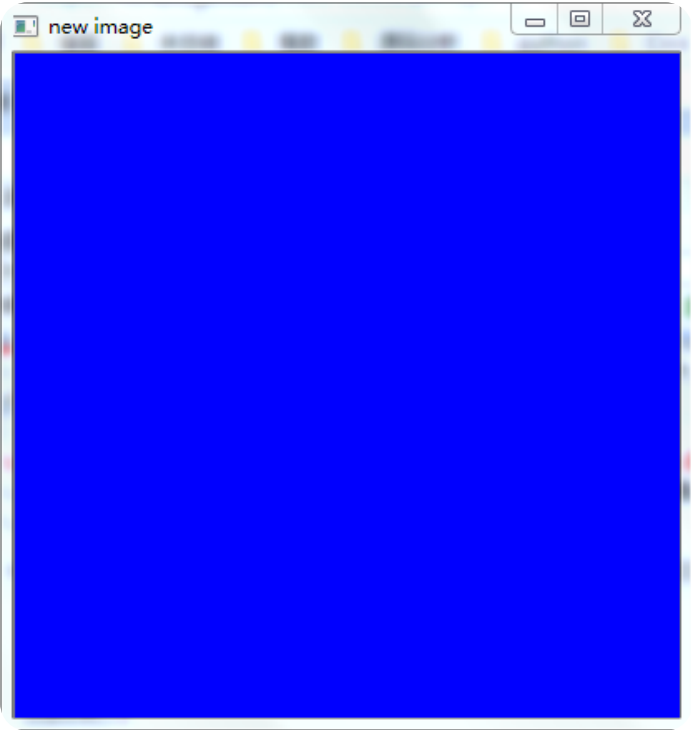
00

关注 | 顶部 | 评论

5. 数据结构（四）树---树的存储结构(1)

推荐排行榜

- 1. 数据结构（七）排序---堆排序(11)
- 2. python---aiohttp的使用(6)
- 3. python---websocket的使用(4)
- 4. python---基础知识回顾（九）图形用户界面-----wxPython(3)
- 5. OpenCV---图像金字塔原理(3)



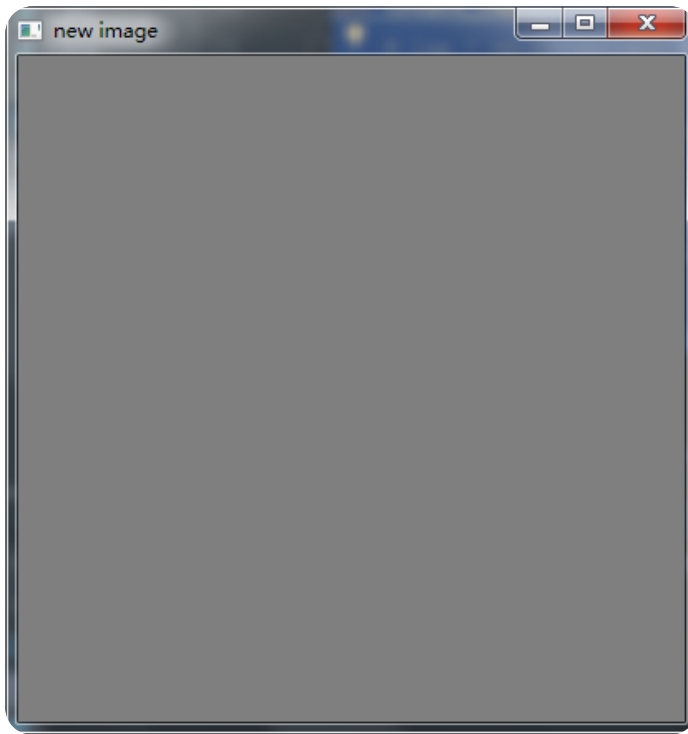
（一）使用多个信道创建图片

```
def create_img():  
    img = np.zeros([400,400,3],np.uint8)    #创建一个三维数组高400，宽400，信号通道3个，初始都为0，每通道占8位个  
    img[:, :, 0] = np.ones([400,400])*255    #将0号通道下[400,400]面积使用ones设置为1，之后乘以255，将其设置为255，注意：3个信道分别是b,g,r所以这里显示为蓝色  
  
    cv.imshow("new image",img)  
  
create_img()  
cv.waitKey(0)    #等待用户操作，里面等待参数是毫秒，我们填写0，代表是永远，等待用户操作  
cv.destroyAllWindows()    #销毁所有窗口
```

（二）使用单个信道创建图像（灰度图像）

00

关注 | 顶部 | 评论



```
def create_img():  
    img = np.zeros([400,400,1],np.uint8)    #创建一个只有一个信道的三维数组，初始为0  
    img[:, :, 0] = np.ones([400,400])*127    #修改这个图像的信道为127，灰色  
    cv.imshow("new image",img)
```

或者（所以初始时候使用ones会更加灵活）

```
def create_img():  
    img = np.ones([400,400,1],np.uint8)    #创建一个只有一个信道的三维数组，初始为1  
    img = img * 127    #可以直接进行运算  
  
    cv.imshow("new image",img)  
    cv.imwrite(".3.png",img)    #可以进行保存
```

### 三：补充Numpy的使用

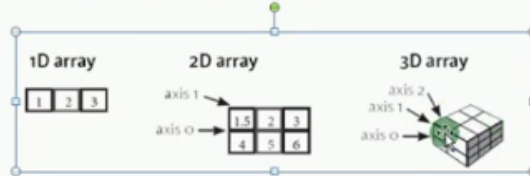
0

0

[关注](#) | [顶部](#) | [评论](#)

## Numpy数组操作

- ▶ Numpy包介绍-[www.numpy.org](http://www.numpy.org)
- ▶ 遍历数组中的每个像素点
- ▶ 修改数组中像素点的值
- ▶ data\dtype\size\shape\len



### (一) 二维数组的使用（选择正确的类型）

#### 1.float类型



```
def create_arr():
    m1 = np.ones([3,3],np.float32)    #float类型，允许小数存在
    m1.fill(122.388)
    print(m1)
```

```
create_arr()
```



```
[[122.388 122.388 122.388]
 [122.388 122.388 122.388]
 [122.388 122.388 122.388]]
```

#### 2.int类型



```
def create_arr():
    m1 = np.ones([3,3],np.uint8)    #不允许小数的存在,且有最大是255
    m1.fill(122.388)
    print(m1)
```

```
create_arr()
```



```
[[122 122 122]
 [122 122 122]
 [122 122 122]]
```



```
def create_arr():
    m1 = np.ones([3,3],np.uint8)    #有位数限制，高位被截断了，低位留下
    m1.fill(256.388)
```

0

0

[关注](#) | [顶部](#) | [评论](#)



```
print(m1)
```

```
create_arr()
```



```
[[0 0 0]
 [0 0 0]
 [0 0 0]]
```

## (二) 维数转换reshape

```
def create_arr():
    m1 = np.ones([3,3],np.uint8)
    m1.fill(122.388)
    m2 = m1.reshape([1,9])    #注意：转换维度，数组大小还是要一致的，不然报错
    print(m2)
```

```
[[122 122 122 122 122 122 122 122 122]]
```

## (三) 使用array自定义数组

```
def create_arr():
    m3 = np.array([[2,2,3],[4,5,6],[7,8,9]],np.uint8)
    print(m3)
```

```
[[2 2 3]
 [4 5 6]
 [7 8 9]]
```

作者：山上有风景

欢迎任何形式的转载，但请务必注明出处。

限于本人水平，如果文章和代码有表述不当之处，还请不吝赐教。

分类： OpenCV ， Python

好文要顶

关注我

收藏该文



山上有风景

关注 - 19

粉丝 - 170

+加关注

« 上一篇： OpenCV---图像加载与保存

» 下一篇： OpenCV---色彩空间（一）

posted @ 2018-07-03 17:59 山上有风景 阅读(2222) 评论(0) 编辑 收藏

0

0

返回顶部

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)， [访问](#) 网站首页。

[关注](#) | [顶部](#) | [评论](#)

Copyright © 2019 山上有风景

Powered by .NET Core 3.1.0 on Linux

0

0

[关注](#) | [顶部](#) | [评论](#)