



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

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)

OpenCV---直线检测

目录

直线检测相关

- Opencv学习笔记-----霍夫变换直线检测及原理解
- OpenCV-Python教程（9、使用霍夫变换检测直线）
- 变换图示

霍夫直线检测的两种方法

- 一：HoughLines霍夫变换

相关知识补充

- （一）HoughLines方法
- 二：HoughLinesP概率霍夫变换（是加强版）使用简单，效果更好，检测图像中分段的直线（而不是贯穿整个图像的直线）

相关知识补充：

- （一）HoughLinesP方法

直线检测相关

Opencv学习笔记-----霍夫变换直线检测及原理解

OpenCV-Python教程（9、使用霍夫变换检测直线）

Hough变换是经典的检测直线的算法。其最初用来检测图像中的直线，同时也可以将其扩展，以用来检测图像中简单的结构。

10

关注 | 顶部 | 评论

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
@ 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，
麻烦问一下如何解决呢？ 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
1. python---websocket的使用(17
253)

直线检测

▶ 霍夫直线变换介绍

▶ 相关API代码演示

霍夫直线变换介绍

▶ **Hough Line Transform**用来做直线检测

▶ 前提条件 - 边缘检测已经完成

▶ 平面空间到极坐标空间转换

霍夫直线变换介绍

$$y = \left(\frac{-\cos \theta}{\sin \theta} \right) x + \left(\frac{r}{\sin \theta} \right)$$

所有都交于一点，由角度值
就可以求出直线方程了

$$r = x \cos \theta + y \sin \theta$$
$$r_0 = x_0 \cdot \cos \theta + y_0 \cdot \sin \theta$$

变换图示

1

0

关注 | 顶部 | 评论

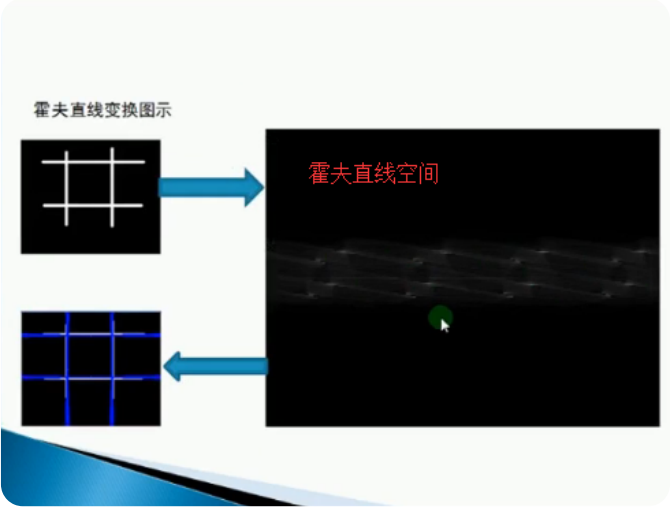
- 2. OpenCV---图像二值化(12513)
- 3. OpenCV---模板匹配matchTemplate(11534)
- 4. OpenCV---直线检测(8208)
- 5. python---基础知识回顾 (九) 图形用户界面-----WxPython(7986)

评论排行榜

- 1. python---基础知识回顾 (九) 图形用户界面-----Tkinter(4)
- 2. python---websocket的使用(3)
- 3. SDN实验---Ryu的源码分析(3)
- 4. 数据结构 (三) 串---KMP模式匹配算法之获取next数组(2)
- 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. Python图像处理库PIL中图像格式转换(3)



霍夫直线检测的两种方法

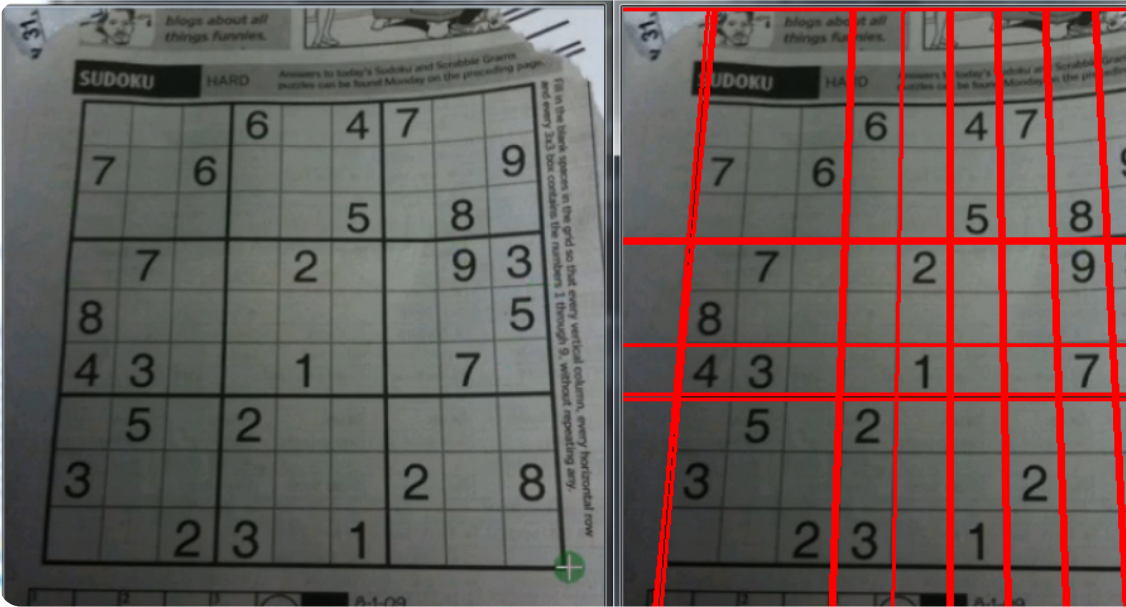
- 1. 获取灰度图像
- 2. canny边缘检测
- 3. 获取霍夫直线信息
- 4. 算出直线位置，画出每条直线

一： HoughLines霍夫变换

```
def line_detection(image):  
    gray = cv.cvtColor(image,cv.COLOR_BGR2GRAY)  
    edges = cv.Canny(gray,50,150,apertureSize=3) #apertureSize是sobel算子大小，只能为1,3,5,7  
    lines = cv.HoughLines(edges,1,np.pi/180,200) #函数将通过步长为1的半径和步长为pi/180的角来搜索所有可能的直线  
    for line in lines:  
        rho,theta = line[0] #获取极值rho长度和theta角度  
        a = np.cos(theta) #获取角度cos值  
        b = np.sin(theta) #获取角度sin值  
        x0 = a * rho #获取x轴值  
        y0 = b * rho #获取y轴值 x0和y0是直线的中点  
        x1 = int(x0 + 1000*(-b)) #获取这条直线最大值点x1  
        y1 = int(y0 + 1000*(a)) #获取这条直线最大值点y1  
        x2 = int(x0 - 1000 * (-b)) #获取这条直线最小值点x2  
        y2 = int(y0 - 1000 * (a)) #获取这条直线最小值点y2 其中*1000是内部规则  
        cv.line(image, (x1,y1), (x2,y2), (0,0,255), 2) #开始划线  
    cv.imshow("image line",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  
  
line_detect_possible_demo(src)  
  
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
```

10

关注 | 顶部 | 评论



相关知识补充

(一) HoughLines方法

```
def HoughLines(image, rho, theta, threshold, lines=None, srn=None, stn=None, min_theta=None, max_theta=None):  
    # real signature unknown; restored from __doc__
```

```
cv.HoughLines(edges,1,np.pi/180,200)
```

cv2.HoughLines函数输出的是[**float**, **float**]形式的ndarray, 其中每个值表示检测到的线(ρ , θ)中浮点值的参数。

第一个参数image: 是canny边缘检测后的图像

第二个参数rho和第三个参数theta: 对应直线搜索的步长。在本例中, 函数将通过步长为1的半径和步长为 $\pi/180$ 的角来搜索所有可能的直线。

最后一个参数threshold: 是经过某一点曲线的数量的阈值, 超过这个阈值, 就表示这个交点所代表的参数对(rho, theta)在原图像中为一条直线

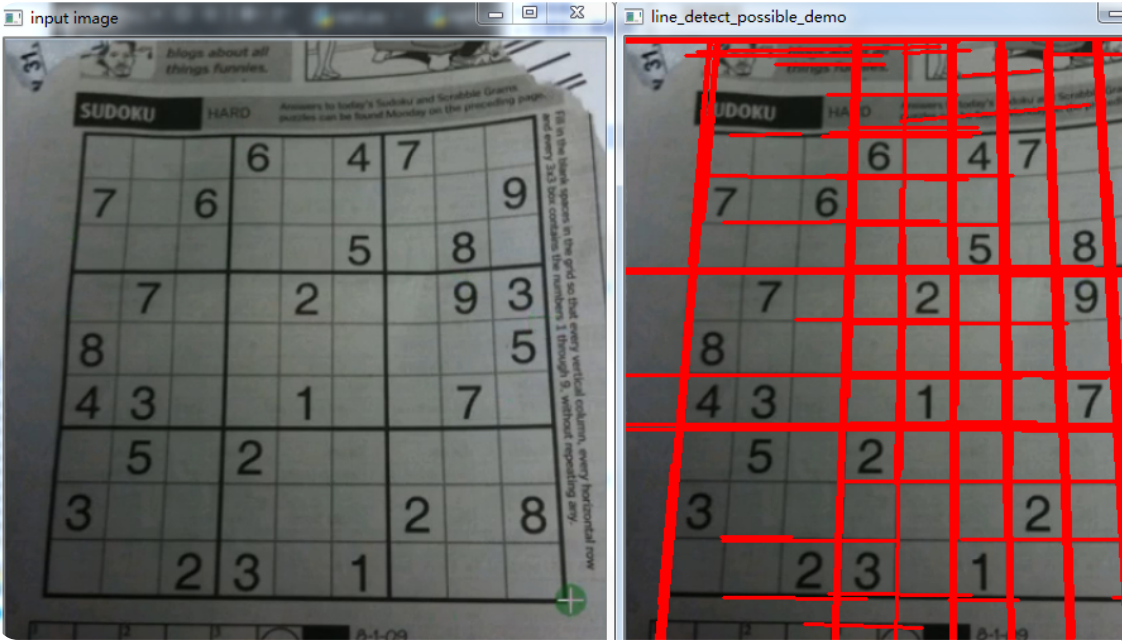
观察前面的例子得到的结果图片, 其中Hough变换看起来就像在图像中查找对齐的边界像素点集合。
但这样会在一些情况下导致虚假检测, 如像素偶然对齐或多条直线穿过同样的对齐像素造成的多重检测。

二: HoughLinesP概率霍夫变换(是加强版)使用简单, 效果更好, 检测图像中分段的直线(而不是贯穿整个图像的直线)

```
def line_detect_possible_demo(image):  
    gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)  
    edges = cv.Canny(gray, 50, 150, apertureSize=3) # apertureSize是sobel算子大小, 只能为1,3,5, 7  
    lines = cv.HoughLinesP(edges, 1, np.pi / 180, 100, minLineLength=50,maxLineGap=10) # 函数将通过步长为1的半径  
    和步长为 $\pi/180$ 的角来搜索所有可能的直线  
    for line in lines:  
        print(type(line)) #多维数组  
        x1,y1,x2,y2 = line[0]  
        cv.line(image,(x1,y1),(x2,y2),(0,0,255),2)  
    cv.imshow("line_detect_possible_demo",image)
```

1 0

关注 | 顶部 | 评论



相关知识补充:

(一) HoughLinesP方法

```
def HoughLinesP(image, rho, theta, threshold, lines=None, minLineLength=None, maxLineGap=None): # real
signature unknown; restored from __doc__
```

```
cv.HoughLinesP(edges, 1, np.pi / 180, 100, minLineLength=50,maxLineGap=10)
```

第一个参数是需要处理的原图像，该图像必须为canny边缘检测后的图像；

第二和第三参数：步长为1的半径和步长为 $\pi/180$ 的角来搜索所有可能的直线

第四个参数是阈值，概念同霍夫变换

第五个参数：minLineLength-线的最短长度，比这个线短的都会被忽略。

第六个参数：maxLineGap-两条线之间的最大间隔，如果小于此值，这两条线就会被看成一条线。

这个函数的返回值就是直线的起点和终点。

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

好文置顶

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---Canny边缘提取
» 下一篇： OpenCV---圆检测

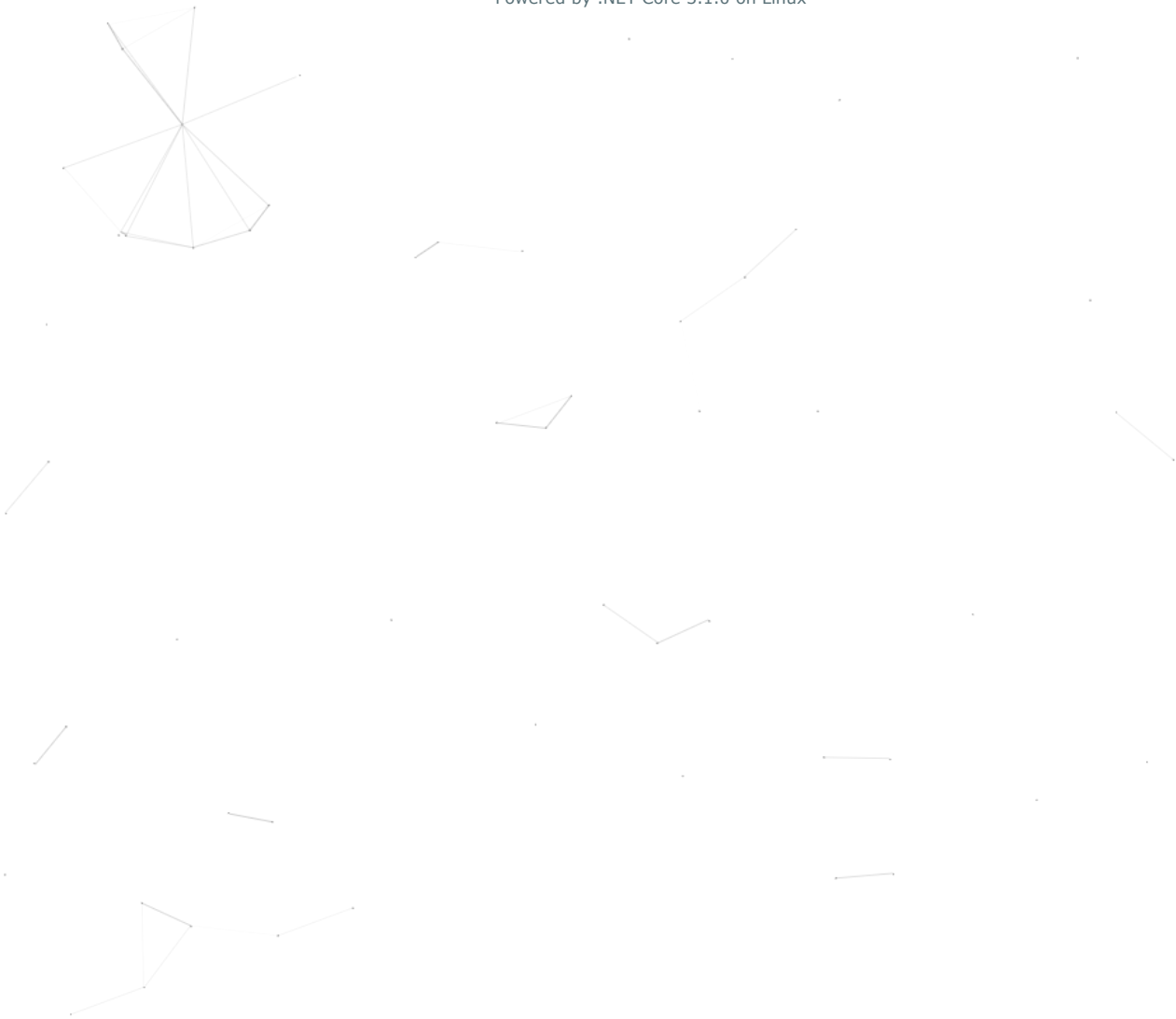
10

关注 | 顶部 | 评论 | 收藏

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。

Copyright © 2019 山上有风景

Powered by .NET Core 3.1.0 on Linux



1 0

[关注](#) | [顶部](#) | [评论](#)