opencv中的一些知识点

1. 坐标系： opencv中的坐标系，坐标原点在左上角，x轴是图像最上面的那条水平线， y轴是图像最左边的垂直线，

numpy中的坐标系，坐标原点在左上角，x轴是图像最左边的那条垂直

y轴是图像最上面的水平线，

1. 图像在计算机中是以矩阵的形式存储，一张宽度 640像素，长度480像素的图像，保存在一个 480 \* 640的矩阵中，即opencv中

Opencv的笔记

目录：

一、GUI特性：

1. 图片的读入、显示、保存；

* 图片读入： im = cv2.imread(imgFile,mode=1)

Mode表示读取图片的形式，有三种

cv2.IMREAD\_COLOR 1 默认参数，表示加载彩色图像BGR格式

cv2.IMREAD\_GRAYSCALE 0 加载灰度图像

cv2.IMREAD\_UNCHANGED -1 表示加载未修改的图像，包括alpha通道参数

* 图片保存： cv2.imwrite(“result.jpg”,img)

前一个参数为保存的文件名，

后一个参数是要保存的图像矩阵

* 图片显示：创建画布-->显示图像-->显示时间-->销毁窗口

cv2.namedWindow(“windowName”,cv2.WINDOW\_NORMAL) 创建画布

cv2.imshow(“windowName”,img) 显示图像

cv2.waitKey(0)

cv2.destoryAllWindows()

其中cv2.namedWindow(窗口标题，参数)，根据参数的不同，可以修改画布的属性，比如窗口大小、比例等，参数设置如下：

窗口大小可改变 cv2.WINDOW\_NORMAL或cv2.WINDOW\_GUI\_NORMAL

窗口大小不可改变 cv2.WINDOW\_AUTOSIZE

窗口大小自适应比例 cv2.WINDOW\_FREERATIO

窗口大小保持比例 cv2.WINDOW\_KEEPRATIO

显示色彩变为暗色 cv2.WINDOW\_GUI\_EXPANDED

cv2.waitKey(delaytime) --->return asciiValue 顾名思义 等待键盘输入的时间，单位为你毫秒ms，即等待指定的毫秒数看是否有键盘输入，按键盘，返回相应按键的ASCII码值，若在delaytime时间内未按键盘，则返回-1，当delaytime为0时，表示永不退出，一直显示，直到有按键按下才会结束waitKey指令。一般会使用如下代码 实现按键退出功能 ：

if cv2.wiatKey(1) & 0xFF == ord(“q”): break

代码解释：等待1ms，检测是否有按键按下，按下的健是否是“q”，ord(“q”)是返回“q”对应的ASCII码，

为什么有0xFF? 这是因为waitKey()返回的值不仅仅是ASCII码，还包含别的码值，因此这里使用的0xFF是一个位掩码，将返回值的左边的24位设置为0，即将返回值限制在0-255之间，

ASCII码 A-Z 65-90 a-z 97-122 回车键Enter--13 退出键ESC--27

1. 视频的捕获、播放、制作（保存），视频文件的属性；

* 捕获（读取）视频 使用cv2.VideoCapture(视频参数) 类

读取流程： 首先创建一个VideoCapture类，然后通过类的方法对视频进行操作，最后释放捕获视频的资源，并清除所有显示窗口

cap = cv2.VideoCapture(0) 创建视频捕获类对象

while cap.isOpened():

ret, frame = cap.read()

cv2.imshow(“frame”,gray)

if cv2.waitKey(1) & 0xFF == ord(“q”):

Break

cap.close()

cv2.destroyAllWindows()

使用cv2.VideoCapture(参数)来初始化类，返回对象cap，参数如果是视频文件地址，则从视频文件中读取视频，参数如果是数字，则表示从指定索引的摄像头读取视频，0表示默认的内置摄像头

cap.isOpended() 检查是否成功初始化，即视频文件或摄像头是否打开，成功返回True，否则返回False

ret, frame = cap.read() 读取视频的一帧信息， ret为True表示读取成功，False表示读取失败， frame表示读取的一帧图像矩阵，

* 视频制作(保存) 使用cv2.VideoWriter(参数)类 创建一个可写的视频对象，

首先确定视频制作的格式 fourcc = cv2.VideoWriter\_fourcc(\*’XVID’) \*”MJPG”

创建可写的视频对象 out = cv2.VideoWriter(视频名，视频格式，帧率，（宽，高）)

将获取的图像帧写入视频 out.write(frame)

释放制作视频所占的资源 out.release()

* 视频属性：使用cap.get() 来获取， 使用cap.set() 来设置

|  |  |
| --- | --- |
| 0 | 视频文件的***当前位置（播放）以毫秒为单位*** |
| 1 | 基于以0开始的被捕获或解码的***帧索引*** |
| 2 | 视频文件的***相对位置***（播放）：0=电影开始，1=影片的结尾。 |
| 3 | 在视频流的帧的***宽度*** |
| 4 | 在视频流的帧的***高度*** |
| 5 | ***帧速率*** |
| 6 | 编解码的4字-字符代码 |
| 7 | 视频文件中的***总帧数*** |
| 8 | 返回对象的格式 |
| 9 | 返回后端特定的值，该值指示当前捕获模式 |
| 10 | 图像的亮度(仅适用于照相机) |
| 11 | 图像的对比度(仅适用于照相机) |
| 12 | 图像的饱和度(仅适用于照相机) |
| 13 | 色调图像(仅适用于照相机) |
| 14 | 图像增益(仅适用于照相机)（Gain在摄影中表示白平衡提升） |
| 15 | 曝光(仅适用于照相机) |
| 16 | 指示是否应将图像转换为RGB布尔标志 |
| 17 | × 暂时不支持 |
| 18 | 立体摄像机的矫正标注（目前只有DC1394 v.2.x后端支持这个功能） |

1. 绘图：直线、矩形、圆、椭圆、多边形和在图片上添加文字；

* 直线 cv2.line(img, pt1, pt2, color, thickness=None,lineType=None, shift=None)

参数： img ---> 源图像，

pt1 ---> 起点坐标， pt2 ---> 终点坐标，

color ---> 线条颜色， BGR颜色元组 (255,0,0) 代表纯蓝色

thickness ---> 线条宽度， 如果给一个闭合曲线设置为-1，则图像会被填充，默认宽度是1；

lineType ---> 线条类型 如8连接，抗锯齿(cv2.LIEN\_AA)等，默认是8连接，

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

* 矩形 cv2.rectangle(img, pt1, pt2, color, thickness=None, lineType=None,shift=None)

pt1,pt2分别代表左上角顶点和右下角顶点的坐标

* 圆形 cv2.circle(img, center, radius, color, thickness=None, lineType=Noe, shift=None)

center 代表圆的中心点坐标

Radius 代表圆的半径大小

* 椭圆形 cv2.ellispe(img, center, axes, angle, startAngle, endAngle, color,

thickness=None, lineType=None, shift=None)

Center 代表椭圆的中心点坐标

axes 是一个2个元素的元组，分别代表长轴和短轴的长度，如(100,50)

angle 是椭圆沿逆时针方向旋转的角度

startAngle 和 endAngle 分别代表椭圆弧沿顺时针方向的起始角度和结束角度，如果是0,360，就代表整个椭圆，如0,180，就代表半个椭圆；

* 多边形 cv2.polylines(img, pts, isClosed, color, thickness=None, lineType=None, shift=None) 可以用来画多条线

pts是所画多边形的顶点坐标数组，是一个N行1列的数组，每个元素都是一个顶点坐标，即行数就是顶点的数目，因此，pts的形状是N\*1\*2

isClosed 如果是False，我们得到的多边形是不闭合的，即首尾不相连，

* 添加文字 cv2.putText(img, text, org, fontFace, fontScale, color, thickness=None, lineType=None)

text 是添加的文字，用字符串表示，目前支持英文，中文支持不友好

org 是文字左上角的坐标，元组表示，(x,y)

fontFace 是字体，如 font = cv2.FONT\_HERSHEY\_SIMPLEX

fontScale 字体大小，用数字表示

1. 画笔：设置鼠标回调函数 cv2.setMouseCallback(windowName, onMouse,param=None)

* 鼠标事件回调函数：所有的鼠标事件回调函数都有同一个格式，如下指令格式，

设置回调函数cv2.setMouseCallback(windowName, onMouse,param=None)，

WindowName是显示窗口名称

onMouse: 是鼠标响应处理函数，即由我们自己写的回调函数，

相当于一直在后台检测鼠标事件，如果在窗口windowName上有鼠标事件发生，则调用回调函数onMouse，在onMouse中根据不同的鼠标事件来执行相应的操作。

其中回调函数onMouse有一个统一的程序格式，如下：

onMouse(event, x, y, flags, param)

onMouse是自定义的回调函数名称，如drawCircle(), drawLine()等

event 是传入到回调函数中的鼠标事件的类型, CV\_EVENT\_\*\*类型

x,y 代表鼠标指针的在图像坐标系中的位置

flags 是CV\_EVENT\_FLAG的组合，一些标志位

Param 是由cv2.setMouseCallback()函数中传递给onMouse()的一些参数

* 常见的鼠标事件种类：

|  |  |  |
| --- | --- | --- |
| 0 | CV\_EVENT\_MOUSEMOVE | 滑动 |
| 1 | EVENT\_LBUTTONDOWN | 左键点击 |
| 2 | EVENT\_RBUTTONDOWN | 右键点击 |
| 3 | EVENT\_MBUTTONDOWN | 中间键点击 |
| 4 | EVENT\_LBUTTONUP | 左键释放 |
| 5 | EVENT\_RBUTTONUP | 右键释放 |
| 6 | EVENT\_MBUTTONUP | 中间键释放 |
| 7 | EVENT\_LBUTTONDBLCLK | 左键双击 |
| 8 | EVENT\_RBUTTONDBLCLK | 右键双击 |
| 9 | EVENT\_MBUTTONDBLCLK | 中间键双击 |

常见的flag

|  |  |  |
| --- | --- | --- |
| 1 | EVENT\_FLAG\_LBUTTON | 左键拖曳 |
| 2 | EVENT\_FLAG\_RBUTTON | 右键拖曳 |
| 4 | EVENT\_FLAG\_MBUTTON | 中键拖曳 |
| 8 | EVENT\_FLAG\_CTRLKEY | 按Ctrl键不放事件 |
| 16 | EVENT\_FLAG\_SHIFTKEY | 按shift不放事件 |
| 32 | EVENT\_FLAG\_ALTKEY | 按ALT不放事件 |

1. 滑动条 cv2.getTrackbarPos() 和 cv2.createTrackbar()等函数

* cv2.createTrackbar(trackbarName, windowName, value, count, onChange)

创建一个可以调整数值的滑动条，并将滑动条绑定到指定的窗口

tarckbarName 滑动条的名称，用于表示创建的滑动条

windowName 窗口名称，表示滑动条会依附在哪个窗口上

Value 表示滑动块的位置，创建时，滑块的初始位置就是该变量的位置，

Count 表示滑块可以达到的最大位置，

onChange 设置滑动条的回调函数，传递的参数是滑动条的位置，即value值

* cv2.getTrackbarPos(trackbarName, windowName)

获取指定滑动条的滑块的当前位置

二、核心操作：

1. 基本操作：
2. 获取/修改像素值，
3. 图像属性，
4. ROI，
5. 拆分/合并图像通道，
6. 图像的扩边
7. 图像运算：
8. 图像加法；
9. 图像混合；
10. 按位运算；

三、图像处理：

1. 颜色空间转换；
2. 几何变换：
3. 扩展缩放；
4. 平移；
5. 旋转；
6. 仿射变换；
7. 透视变换；
8. 图像阈值处理：
9. 简单阈值；
10. 自适应阈值；
11. Otsu 最大方差阈值；
12. 图像平滑处理：
13. 平均；
14. 高斯模糊；
15. 中值模糊；
16. 双边模糊；
17. 图像形态学处理：
18. 膨胀、腐蚀；
19. 开、闭运算；
20. 形态学梯度；
21. 礼帽、黑帽；
22. 图像梯度：Sobel算子、Scharr算子、Laplacian算子；
23. 边缘检测：canny边缘检测；
24. 图像金字塔： 图像融合
25. 轮廓：
26. 获取、绘制轮廓
27. 轮廓的特征： 矩、轮廓面积、轮廓周长、轮廓近似、凸包检测、边界矩形、最小外接圆、椭圆拟合、直线拟合；
28. 轮廓的性质：长宽比、Extent、Solidity、方向、掩模和像素点、最大轮廓和最小轮廓、平均颜色和平均灰度、极点；
29. 轮廓的层次结构：
30. 直方图：
31. 统计直方图、绘制直方图、使用掩模的直方图；
32. 直方图均衡化；
33. 2D直方图；
34. 直方图 反向投影；
35. 图像变换： 傅里叶变换
36. Hough 直线变换和圆环变换
37. 模板匹配
38. 图像特征提取和描述
39. Harris角点检测
40. SIFT scale-invariant feature transform 尺度不变特征转换
41. SURF Speeded-Up Robust Feature 加速的
42. FAST
43. ORB