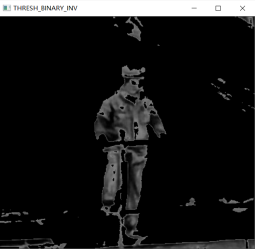
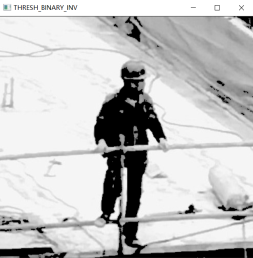
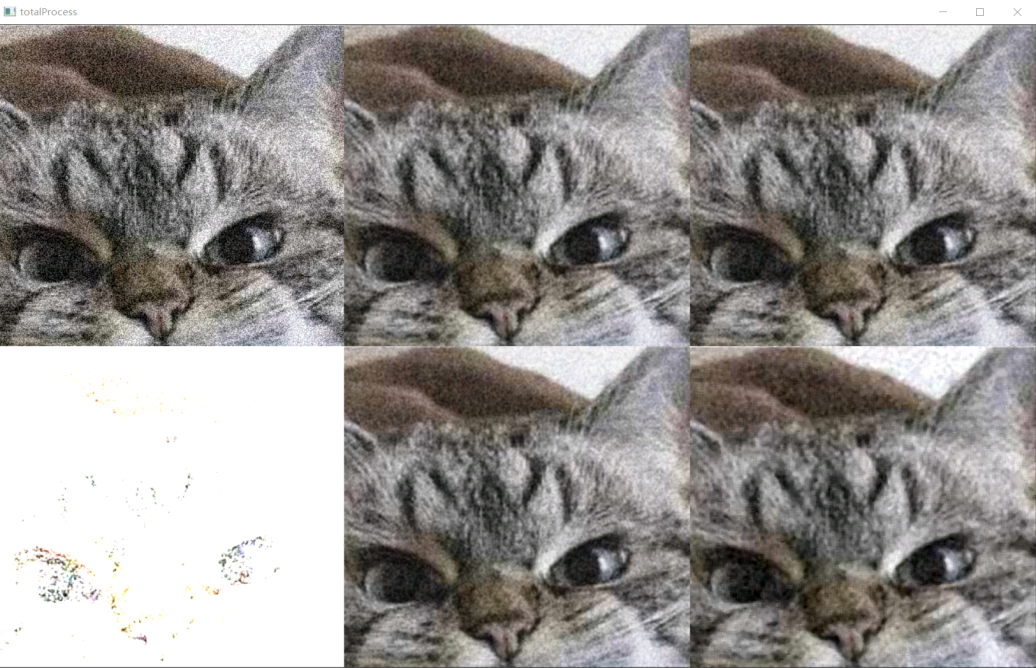
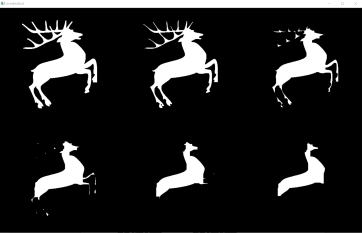
*'''  
2021.1.29FromIvicxDS:openCV;E2阈值函数  
'''***'''  
5种阈值处理  
'''**import cv2*#读取格式为BGR*def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
img=cv2.imread(**"image/P1.jpg"**,cv2.IMREAD\_GRAYSCALE)  
*#图像，阈值，max  
#有两个返回值，一般只用第二个*ret,thresh1=cv2.threshold(img,127,255,cv2.THRESH\_BINARY)*#>127->255,<127->0*ret,thresh2=cv2.threshold(img,127,255,cv2.THRESH\_BINARY\_INV)*#1的反转黑白颠倒*ret,thresh3=cv2.threshold(img,127,255,cv2.THRESH\_TRUNC)*#设置截断值>127->127else不变*ret,thresh4=cv2.threshold(img,127,255,cv2.THRESH\_TOZERO)*#设置截断值>127不变else->0*ret,thresh5=cv2.threshold(img,127,255,cv2.THRESH\_TOZERO\_INV)  
showPicture(**"Original Picture"**,img)  
showPicture(**"THRESH\_BINARY"**,thresh1)  
showPicture(**"THRESH\_BINARY\_INV"**,thresh2)  
showPicture(**"THRESH\_BINARY\_INV"**,thresh3)  
showPicture(**"THRESH\_BINARY\_INV"**,thresh4)  
showPicture(**"THRESH\_BINARY\_INV"**,thresh5)



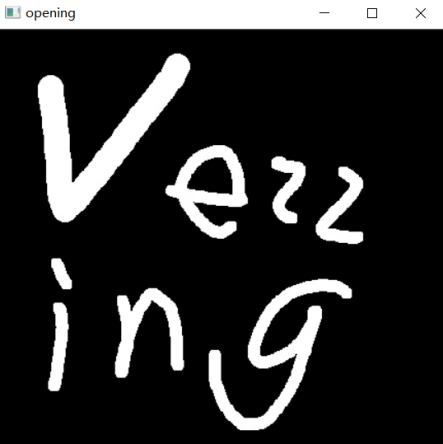
*'''  
2021.1.29FromIvicxDS:openCV;E2平滑处理  
'''***'''  
平均卷积操作【均值滤波】  
构造卷积矩阵求内积  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
img=cv2.imread(**"image/Noise.jpg"**)  
showPicture(**"OriginalPicture"**,img)  
  
*#均值滤波\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# □□□  
# □■□  
# □□□*blur=cv2.blur(img,(3,3))*#指定卷积矩阵的大小，通常是奇数【中心点】*showPicture(**"BluredPicture"**,blur)  
  
*#方框滤波\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**box1=cv2.boxFilter(img,-1,(3,3),normalize=True)  
*#执行归一化(/9)结果与均值滤波相同*showPicture(**"BoxNormalized"**,box1)  
  
box2=cv2.boxFilter(img,-1,(3,3),normalize=False)  
*#不执行归一化(/9)结果直接相加越界当做255*showPicture(**"BoxInnormalized"**,box2)  
  
*#高斯滤波\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
#\_/\\_高斯函数***"""  
[0.6,0.8,0.6  
 0.8,1 ,0.8  
 0.6,0.8,0.6]  
按照距离赋予权重  
"""**aussian=cv2.GaussianBlur(img,(5,5),1)  
showPicture(**"GaussianBlur"**,aussian)  
  
*#中值滤波\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
#矩阵内的数从小到大排序去中间值*median=cv2.medianBlur(img,5)  
showPicture(**"medianBlur"**,median)*#当有噪音点时可以完全过滤*res1=numpy.hstack((img,blur,box1))  
res2=numpy.hstack((box2,aussian,median))  
res=numpy.vstack((res1,res2))  
showPicture(**"totalProcess"**,res)



*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：腐蚀操作1  
'''***'''  
1 1 1  
1 0 1  
0 0 0  
->  
1 1 1  
1 1 1  
1 1 1   
框内有黑色则改为全黑  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
img=cv2.imread(**"image/kernel.jpg"**)  
showPicture(**"originalPicture"**,img)  
  
*#腐蚀操作\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**kernel=numpy.ones((3,3),numpy.uint8)  
erosion=cv2.erode(img,kernel,iterations=1)  
showPicture(**"eroded"**,erosion)  
  
dear=cv2.imread(**"image/kernel2.jpg"**)  
erosion1=cv2.erode(dear,kernel,iterations=1)  
erosion2=cv2.erode(dear,kernel,iterations=3)  
erosion3=cv2.erode(dear,kernel,iterations=5)  
erosion4=cv2.erode(dear,kernel,iterations=7)  
erosion5=cv2.erode(dear,kernel,iterations=9)  
  
res1=numpy.hstack((dear,erosion1,erosion2))  
res2=numpy.hstack((erosion3,erosion4,erosion5))  
res=numpy.vstack((res1,res2))  
  
showPicture(**"erodedList"**,res)

*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：膨胀操作1  
'''***'''  
1 0 1  
1 0 0  
0 0 0  
->  
0 0 0  
0 0 0  
0 0 0  
框内有白色则改为全白  
先腐蚀再膨胀可以去除杂边  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
img=cv2.imread(**"image/kernel.jpg"**)  
showPicture(**"originalPicture"**,img)  
  
*#腐蚀操作\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**kernel=numpy.ones((3,3),numpy.uint8)  
erosion=cv2.erode(img,kernel,iterations=3)  
showPicture(**"eroded"**,erosion)  
  
*#膨胀操作\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**process=cv2.dilate(erosion,kernel,iterations=3)  
showPicture(**"returned"**,process)  
  
res=numpy.hstack((img,erosion,process))  
showPicture(**"totalProcess"**,res)  
  
**'''  
开运算：先腐蚀再膨胀  
闭运算：先膨胀再腐蚀  
'''**

*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：开运算  
'''*import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
img=cv2.imread(**"image/kernel.jpg"**)  
showPicture(**"originalPicture"**,img)  
  
kernel=numpy.ones((5,5),numpy.uint8)  
opening=cv2.morphologyEx(img,cv2.MORPH\_OPEN,kernel)  
  
showPicture(**"opening"**,opening)

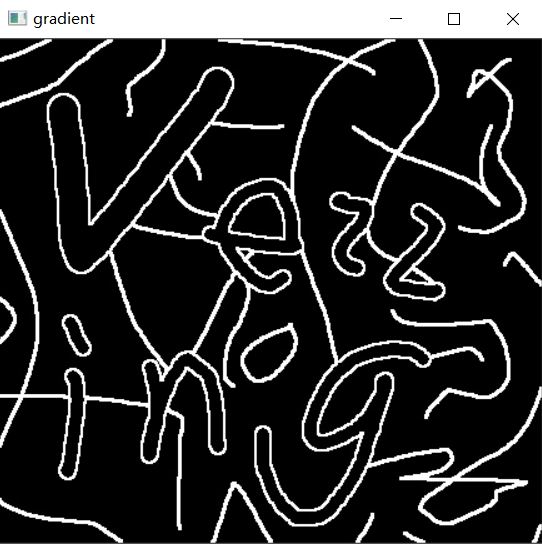


*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：闭运算  
'''*import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
img=cv2.imread(**"image/kernel.jpg"**)  
showPicture(**"originalPicture"**,img)  
  
kernel=numpy.ones((5,5),numpy.uint8)  
close=cv2.morphologyEx(img,cv2.MORPH\_CLOSE,kernel)  
  
showPicture(**"close"**,close)

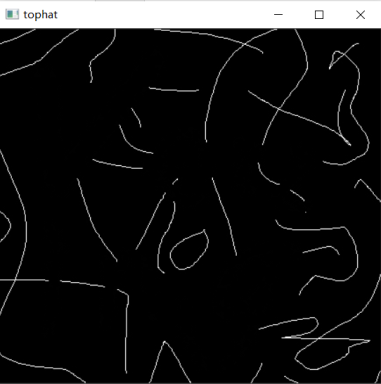


*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：梯度运算  
'''***'''  
对图像  
膨胀  
腐蚀  
膨胀结果-腐蚀结果->轮廓信息【膨胀腐蚀后一样的地方(图形内,背景)相减后为0】  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
kernel=numpy.ones((3,3),numpy.uint8)  
img=cv2.imread(**"image/kernel.jpg"**)  
  
dilate=cv2.dilate(img,kernel,iterations=1)  
erosion=cv2.erode(img,kernel,iterations=1)  
  
result=dilate-erosion  
res1=numpy.hstack((img,result))  
res2=numpy.hstack((dilate,erosion))  
res=numpy.vstack((res1,res2))  
showPicture(**""**,res)

*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：梯度运算cv2函数  
'''***'''  
对图像  
膨胀  
腐蚀  
膨胀结果-腐蚀结果->轮廓信息【膨胀腐蚀后一样的地方(图形内,背景)相减后为0】  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
kernel=numpy.ones((3,3),numpy.uint8)  
img=cv2.imread(**"image/kernel.jpg"**)  
gradient=cv2.morphologyEx(img,cv2.MORPH\_GRADIENT,kernel)  
showPicture(**"gradient"**,gradient)



*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：礼帽运算  
'''***'''  
对图像  
开运算  
原始图像-开运算结果->噪点信息  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
kernel=numpy.ones((3,3),numpy.uint8)  
img=cv2.imread(**"image/kernel.jpg"**)  
tophat=cv2.morphologyEx(img,cv2.MORPH\_TOPHAT,kernel)  
showPicture(**"tophat"**,tophat)



*'''  
2021.1.29FromIvicxDS:openCV;E2图像形态处理：黑帽运算  
'''***'''  
对图像  
闭运算  
闭运算结果-原始图像->  
'''**import cv2*#读取格式为BGR*import numpy  
  
def showPicture(name,picture):  
 *#图像的显示,也可以显示多窗口* cv2.imshow(name,picture)  
 *#在键盘中按任意键退出显示并向后执行语句  
 #cv2.waitKey(1000)表示只显示1秒* cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
kernel=numpy.ones((3,3),numpy.uint8)  
img=cv2.imread(**"image/kernel.jpg"**)  
tophat=cv2.morphologyEx(img,cv2.MORPH\_BLACKHAT,kernel)  
showPicture(**"blackhat"**,tophat)

