[](https://www.csdn.net/)

* [博客](https://blog.csdn.net/)
* [下载](https://download.csdn.net/)
* [学习](https://edu.csdn.net/)
* [社区](https://bbs.csdn.net/)
* [GitCode [https://img-home.csdnimg.cn/images/20221027045535.png](https://gitcode.net/gitcode/1024opensource?utm_source=csdn_toolbar)](https://gitcode.net/gitcode/1024opensource?utm_source=csdn_toolbar)
* [云服务](https://dev-portal.csdn.net/welcome?utm_source=toolbar)
* [猿如意](https://devbit.csdn.net/?source=csdn_toolbar)



 搜索

登录/注册

[会员中心 [https://img-home.csdnimg.cn/images/20210918025138.gif](https://mall.csdn.net/vip)](https://mall.csdn.net/vip)

[足迹](https://i.csdn.net/#/user-center/collection-list?type=1)

[动态](https://blink.csdn.net/)

[创作中心 [https://img-home.csdnimg.cn/images/20220627041202.png](https://mp.csdn.net/)](https://mp.csdn.net/)

[发布](https://mp.csdn.net/edit)

**图片的侵蚀cv2.erode()和膨胀cv2.dilate()操作**



[大梦冲冲冲](https://blog.csdn.net/qq_37844044)https://csdnimg.cn/release/blogv2/dist/pc/img/newCurrentTime2.png于 2021-03-29 09:35:52 发布https://csdnimg.cn/release/blogv2/dist/pc/img/articleReadEyes2.png1147https://csdnimg.cn/release/blogv2/dist/pc/img/tobarCollect2.png 收藏

分类专栏： [python零散知识总结](https://blog.csdn.net/qq_37844044/category_10808021.html) 文章标签： [计算机视觉](https://so.csdn.net/so/search/s.do?q=%E8%AE%A1%E7%AE%97%E6%9C%BA%E8%A7%86%E8%A7%89&t=blog&o=vip&s=&l=&f=&viparticle=) [opencv](https://so.csdn.net/so/search/s.do?q=opencv&t=blog&o=vip&s=&l=&f=&viparticle=) [图像识别](https://so.csdn.net/so/search/s.do?q=%E5%9B%BE%E5%83%8F%E8%AF%86%E5%88%AB&t=blog&o=vip&s=&l=&f=&viparticle=)

版权

[[](https://blog.csdn.net/qq_37844044/category_10808021.html)python零散知识总结专栏收录该内容](https://blog.csdn.net/qq_37844044/category_10808021.html)

21 篇文章0 订阅

订阅专栏

**图片的侵蚀和膨胀操作**

这是cv2中的两个针对图片的简单操作、处理，分别对应函数cv2.erode和cv2.dilate.

**cv2.erode()腐蚀**

将前景物体变小，理解成将图像断开裂缝变大

**cv2.dilate()膨胀**

cv2.dilate()膨胀：将前景物体变大，理解成将图像断开裂缝变小

import cv2

image = cv2.imread(r"23080.jpg")

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

#cv2.imshow("Gray Image", gray)

kernel = cv2.getStructuringElement(cv2.MORPH\_ELLIPSE, (7,7))

dilated = cv2.dilate(gray.copy(), kernel, 10)

# dilated = cv2.dilate(gray.copy(), None, 10)

eroded = cv2.erode(gray.copy(), kernel, 10)

# eroded = cv2.erode(gray.copy(), None, 10)

#cv2.imshow("Eroded Image", eroded)

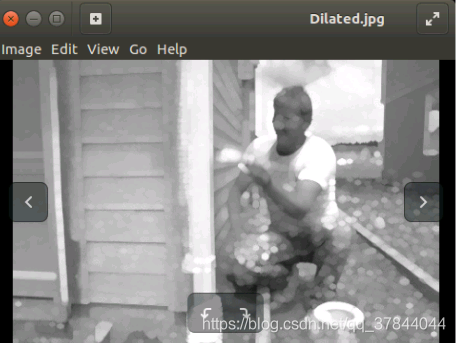
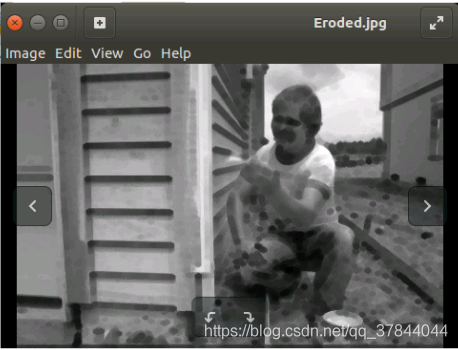
cv2.imwrite('Eroded.jpg',eroded)

cv2.imwrite('Dilated.jpg',dilated)

cv2.imwrite('C.jpg',dilated-eroded)

cv2.waitKey(0)

cv2.destroyAllWindows()

# OpenCV-Python教程：形态学变换~腐蚀和膨胀(erode,dilate)

形态学变换是基于图像形状的变换过程，通常用来处理二值图像，当然也可以用在灰度图上。OpenCV中的形态学变换同平滑处理一样也是基于一种“滑动窗口”的操作，不过在形态学变换中“滑动窗口”有一个更专业的名词：“结构元”，也可以像平滑处理那样称呼为kernel，结构元的形状有方形、十字形、椭圆形等，其形状决定了形态学变换的特点。形态学变换主要有腐蚀、膨胀、开操作、闭操作等等。

## 1、腐蚀erode()

腐蚀操作可以将边界的白色（前景）像素“腐蚀”掉，但仍能保持大部分白色。类似平滑处理的滑动窗口，用某种结构元在图像上滑动，当结构元覆盖原始图像中的所有像素都为“1”时，新图像中该像素点的值才为“1”（CV8U为255）。腐蚀可以用来去除噪声、去掉“粘连”。

接口形式：

cv2.erode(src, kernel[, dst[, anchor[, iterations[, borderType[, borderValue]]]]]) ->dst

* **参数含义：**
* src：通道数任意；图像深度只能是CV\_8U, CV\_16U, CV\_16S, CV\_32F or CV\_64F；
* kernel：可以由getStructuringElement()构建；
* dst：输出图像，通道数和数据类型同src；
* anchor：锚点，默认使用(-1,-1)表示中心点；
* iterations：腐蚀次数；
* borderType：边界类型；
* borderValue：边界值；

下面是一个kernel大小为5和7的腐蚀例子：

import matplotlib.pyplot as plt

import cv2

print('VX公众号: 桔子code / juzicode.com')

print('cv2.\_\_version\_\_:',cv2.\_\_version\_\_)

plt.rc('font',family='Youyuan',size='9')

img = cv2.imread('..\\samples\\picture\\mnist-7.jpg',cv2.IMREAD\_GRAYSCALE)

\_,img = cv2.threshold(img,127,255,0)

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(5,5)) *#ksize=5,5*

img\_ret1 = cv2.erode(img,kernel,iterations=1)

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(7,7)) *#ksize=7x7,*

img\_ret2 = cv2.erode(img,kernel,iterations=1)

img\_ret3 = cv2.erode(img,kernel,iterations=2) *#ksize=7x7，腐蚀2次*

*#显示图像*

fig,ax = plt.subplots(2,2)

ax[0,0].set\_title('原图-mnist(juzicode.com)')

ax[0,0].imshow(cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)) *#matplotlib显示图像为rgb格式*

ax[0,1].set\_title('erode ksize=5')

ax[0,1].imshow(cv2.cvtColor(img\_ret1,cv2.COLOR\_BGR2RGB))

ax[1,0].set\_title('erode ksize=7')

ax[1,0].imshow(cv2.cvtColor(img\_ret2,cv2.COLOR\_BGR2RGB))

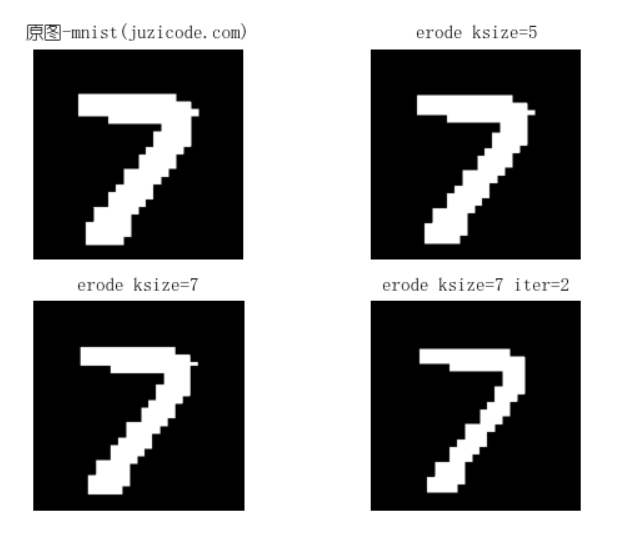
ax[1,1].set\_title('erode ksize=7 iter=2')

ax[1,1].imshow(cv2.cvtColor(img\_ret3,cv2.COLOR\_BGR2RGB))

ax[0,0].axis('off');ax[0,1].axis('off');ax[1,0].axis('off');ax[1,1].axis('off')*#关闭坐标轴显示*

plt.show()

运行结果：

原图图源：mnist

 从运行结果可以看到kernel的ksize越大，iterations次数越多，图像看起来越“廋”。

## 2、膨胀dilate()

膨胀是腐蚀的逆操作，可以将边界的白色（前景）像素“生长”扩大。滑动窗口经过白色像素时，只要结构元中有1个像素为“1”时，新图像中该像素点的值就为“1”（CV8U为255）。膨胀可以用来增强连接、填充凹痕。

接口形式：

cv2.dilate(src, kernel[, dst[, anchor[, iterations[, borderType[, borderValue]]]]]) ->dst

* **参数含义：**
* src：通道数任意；图像深度只能是CV\_8U, CV\_16U, CV\_16S, CV\_32F or CV\_64F；
* kernel：可以由getStructuringElement()构建；
* dst：输出图像，通道数和数据类型同src；
* anchor：锚点，默认使用(-1,-1)表示中心点；
* iterations：膨胀次数；
* borderType：边界类型；
* borderValue：边界值；

下面是一个kernel大小为5和7的膨胀例子：

import matplotlib.pyplot as plt

import cv2

print('VX公众号: 桔子code / juzicode.com')

print('cv2.\_\_version\_\_:',cv2.\_\_version\_\_)

plt.rc('font',family='Youyuan',size='9')

img = cv2.imread('..\\samples\\picture\\mnist-7-b.jpg',cv2.IMREAD\_GRAYSCALE)

\_,img = cv2.threshold(img,127,255,0)

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(5,5))

img\_ret1 = cv2.dilate(img,kernel,iterations=1)

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(7,7))

img\_ret2 = cv2.dilate(img,kernel,iterations=1)

img\_ret3 = cv2.dilate(img,kernel,iterations=2)

*#显示图像*

fig,ax = plt.subplots(2,2)

ax[0,0].set\_title('原图-mnist(juzicode.com)')

ax[0,0].imshow(cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)) *#matplotlib显示图像为rgb格式*

ax[0,1].set\_title('dilate ksize=5')

ax[0,1].imshow(cv2.cvtColor(img\_ret1,cv2.COLOR\_BGR2RGB))

ax[1,0].set\_title('dilate ksize=7')

ax[1,0].imshow(cv2.cvtColor(img\_ret2,cv2.COLOR\_BGR2RGB))

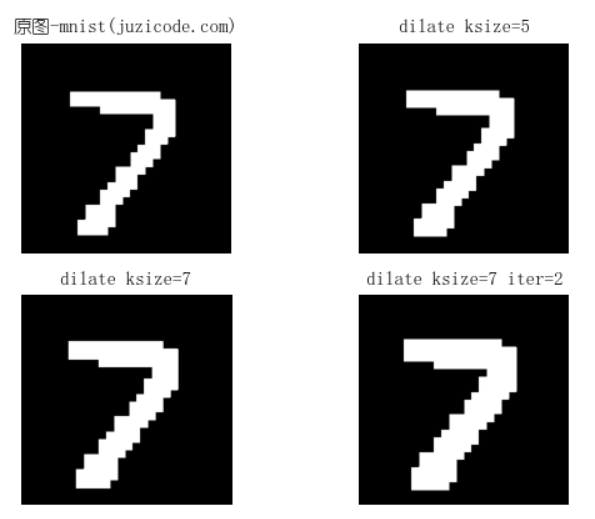
ax[1,1].set\_title('dilate ksize=7 iter=2')

ax[1,1].imshow(cv2.cvtColor(img\_ret3,cv2.COLOR\_BGR2RGB))

ax[0,0].axis('off');ax[0,1].axis('off');ax[1,0].axis('off');ax[1,1].axis('off')*#关闭坐标轴显示*

plt.show()

运行结果：

原图图源：mnist

 从运行结果可以看到kernel的ksize越大，iterations次数越多，图像看起来越“胖”。

## 3、结构元生成getStructuringElement()

结构元生成函数用来生成形态学变换的kernel参数。

接口形式：

cv2.getStructuringElement(shape, ksize[, anchor]) ->retval

* **参数含义：**
* shape：结构元(kernel)的形状；
* ksize：结构元(kernel)的大小；
* anchor：锚点，默认使用(-1,-1)表示中心点；

其中shape属性有可选的3种形状：

|  |  |
| --- | --- |
| cv2.MORPH\_RECT | 方形，所有的数值均为1 |
| cv2.MORPH\_CROSS | 十字交叉形，在锚点坐标的水平和竖直方向的元素为1，其他为0 |
| cv2.MORPH\_ELLIPSE | 椭圆形 |

下面是一个kernel大小为7，形状不同的膨胀例子：

import matplotlib.pyplot as plt

import cv2

print('VX公众号: 桔子code / juzicode.com')

print('cv2.\_\_version\_\_:',cv2.\_\_version\_\_)

plt.rc('font',family='Youyuan',size='9')

img = cv2.imread('..\\samples\\picture\\mnist-7-b.jpg',cv2.IMREAD\_GRAYSCALE)

\_,img = cv2.threshold(img,127,255,0)

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(7,7))

print('MORPH\_RECT kernel:\n',kernel)

img\_ret1 = cv2.dilate(img,kernel,iterations=1)

kernel = cv2.getStructuringElement(cv2.MORPH\_ELLIPSE,(7,7))

print('MORPH\_ELLIPSE kernel:\n',kernel)

img\_ret2 = cv2.dilate(img,kernel,iterations=1)

kernel = cv2.getStructuringElement(cv2.MORPH\_CROSS,(7,7))

print('MORPH\_CROSS kernel:\n',kernel)

img\_ret3 = cv2.dilate(img,kernel,iterations=1)

*#显示图像*

fig,ax = plt.subplots(2,2)

ax[0,0].set\_title('原图-mnist(juzicode.com)')

ax[0,0].imshow(cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)) *#matplotlib显示图像为rgb格式*

ax[0,1].set\_title('MORPH\_RECT kernel')

ax[0,1].imshow(cv2.cvtColor(img\_ret1,cv2.COLOR\_BGR2RGB))

ax[1,0].set\_title('MORPH\_ELLIPSE kernel')

ax[1,0].imshow(cv2.cvtColor(img\_ret2,cv2.COLOR\_BGR2RGB))

ax[1,1].set\_title('MORPH\_CROSS kernel')

ax[1,1].imshow(cv2.cvtColor(img\_ret3,cv2.COLOR\_BGR2RGB))

ax[0,0].axis('off');ax[0,1].axis('off');ax[1,0].axis('off');ax[1,1].axis('off')*#关闭坐标轴显示*

plt.show()

运行结果：

VX公众号: 桔子code / juzicode.com

cv2.\_\_version\_\_: 4.5.3

MORPH\_RECT kernel:

[[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]]

MORPH\_ELLIPSE kernel:

[[0 0 0 1 0 0 0]

[0 1 1 1 1 1 0]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[1 1 1 1 1 1 1]

[0 1 1 1 1 1 0]

[0 0 0 1 0 0 0]]

MORPH\_CROSS kernel:

[[0 0 0 1 0 0 0]

[0 0 0 1 0 0 0]

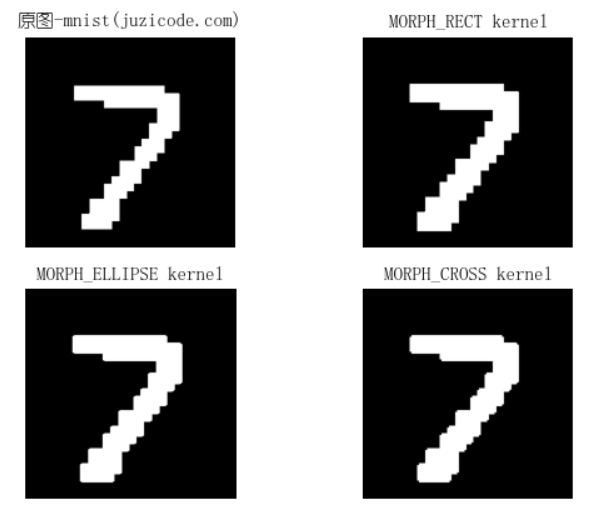
[0 0 0 1 0 0 0]

[1 1 1 1 1 1 1]

[0 0 0 1 0 0 0]

[0 0 0 1 0 0 0]

[0 0 0 1 0 0 0]]

原图图源：mnist

从运行结果可以看到当使用方形MORPH\_RECT的结构元时，新图像的边界看起来仍然是方方正正的，但是使用十字形MORPH\_CROSS和椭圆形MORPH\_ELLIPSE的结构元时，边界要显得“圆滑”的多。

## 4、实际应用的例子

下面是一个用腐蚀去除噪声的例子：

import matplotlib.pyplot as plt

import cv2

print('VX公众号: 桔子code / juzicode.com')

print('cv2.\_\_version\_\_:',cv2.\_\_version\_\_)

plt.rc('font',family='Youyuan',size='9')

img = cv2.imread('..\\samples\\picture\\mnist-7-noise.jpg',cv2.IMREAD\_GRAYSCALE)

\_,img = cv2.threshold(img,127,255,0)

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(7,7))

img\_ret1 = cv2.erode(img,kernel,iterations=2)

*#显示图像*

fig,ax = plt.subplots(1,2)

ax[0].set\_title('原图 (juzicode.com)')

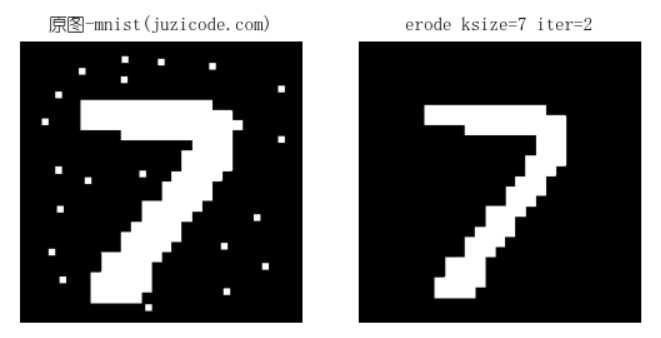
ax[0].imshow(cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)) *#matplotlib显示图像为rgb格式*

ax[1].set\_title('erode ksize=7 iter=2')

ax[1].imshow(cv2.cvtColor(img\_ret1,cv2.COLOR\_BGR2RGB))

ax[0].axis('off');ax[1].axis('off')

plt.show()

原图图源：mnist

在原图中白色小点表示的噪声通过腐蚀操作被去除掉了。

下面是一个在五线谱中去除横线的例子：

import matplotlib.pyplot as plt

import cv2

print('VX公众号: 桔子code / juzicode.com')

print('cv2.\_\_version\_\_:',cv2.\_\_version\_\_)

plt.rc('font',family='Youyuan',size='9')

img = cv2.imread('..\\samples\\data\\notes.png',cv2.IMREAD\_GRAYSCALE)

\_,img\_bin = cv2.threshold(img,127,255,1)*#二值反色*

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT,(1,7))

img\_erode = cv2.erode(img\_bin,kernel,iterations=1)

img\_dilate = cv2.dilate(img\_erode,kernel,iterations=1)

*#显示图像*

fig,ax = plt.subplots(3,1)

ax[0].set\_title('原图 (juzicode.com)')

ax[0].imshow(cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)) *#matplotlib显示图像为rgb格式*

ax[1].set\_title('img\_erode')

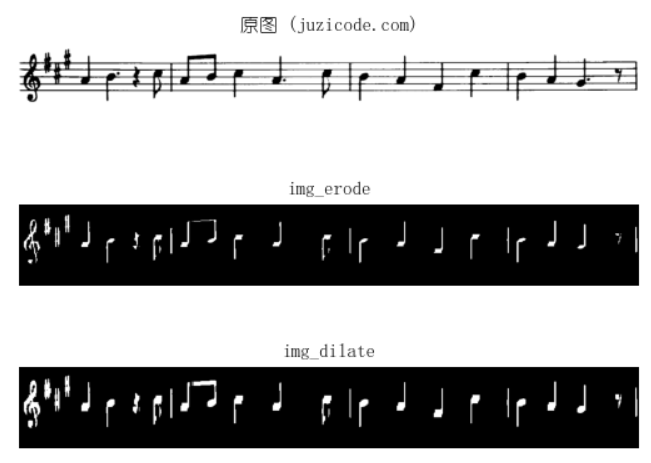
ax[1].imshow(cv2.cvtColor(img\_erode,cv2.COLOR\_BGR2RGB))

ax[2].set\_title('img\_dilate')

ax[2].imshow(cv2.cvtColor(img\_dilate,cv2.COLOR\_BGR2RGB))

ax[0].axis('off');ax[1].axis('off');ax[2].axis('off')

plt.show()



因为横线是水平方向的，高度小长度大，先用一个1×7的结构元进行腐蚀，这时腐蚀会作用到垂直方向上，水平方向几乎不会被影响到，从而将高度较小的横线腐蚀掉，但是因为腐蚀操作将需要保留的符号也腐蚀掉了一部分，所以再用同样的结构元进行一次膨胀操作，就能复原出符号的形状。